

High-Order Finite Element Methods for Moving Boundary Problems with Prescribed Boundary Evolution

Evan S. Gawlik^a, Adrian J. Lew^{b,a}

^a*Computational and Mathematical Engineering, Stanford University*

^b*Mechanical Engineering, Stanford University*

Abstract

We introduce a framework for the design of finite element methods for two-dimensional moving boundary problems with prescribed boundary evolution that have arbitrarily high order of accuracy, both in space and in time. At the core of our approach is the use of a universal mesh: a stationary background mesh containing the domain of interest for all times that adapts to the geometry of the immersed domain by adjusting a small number of mesh elements in the neighborhood of the moving boundary. The resulting method maintains an exact representation of the (prescribed) moving boundary at the discrete level, or an approximation of the appropriate order, yet is immune to large distortions of the mesh under large deformations of the domain. The framework is general, making it possible to achieve any desired order of accuracy in space and time by selecting a preferred and suitable finite-element space on the universal mesh for the problem at hand, and a preferred and suitable time integrator for ordinary differential equations. We illustrate our approach by constructing a particular class of methods, and apply them to a prescribed-boundary variant of the Stefan problem. We present numerical evidence for the order of accuracy of our schemes in one and two dimensions.

Keywords: Moving boundary, universal mesh, free boundary, ALE, Stefan problem

1. Introduction

Science and engineering are replete with instances of moving boundary problems: partial differential equations posed on domains that change with time. Problems of this type, which arise in areas as diverse as fluid-structure interaction, multiphase flow physics, and fracture mechanics, are inherently challenging to solve numerically.

Broadly speaking, computational methods for moving boundary problems generally adhere to one of two paradigms. *Deforming-mesh* methods employ a computational mesh that deforms in concert with the moving domain, whereas *fixed-mesh* methods employ a stationary background mesh in which the domain is immersed. While the former approach can require that efforts be made to avoid distortions of the mesh under large deformations [1], the latter approach requires that special care be taken in order to account for any discrepancy between the exact boundary and element interfaces [2, 3]. Figs. 1-2 illustrate these two

Email addresses: egawlik@stanford.edu (Evan S. Gawlik), lewa@stanford.edu (Adrian J. Lew)

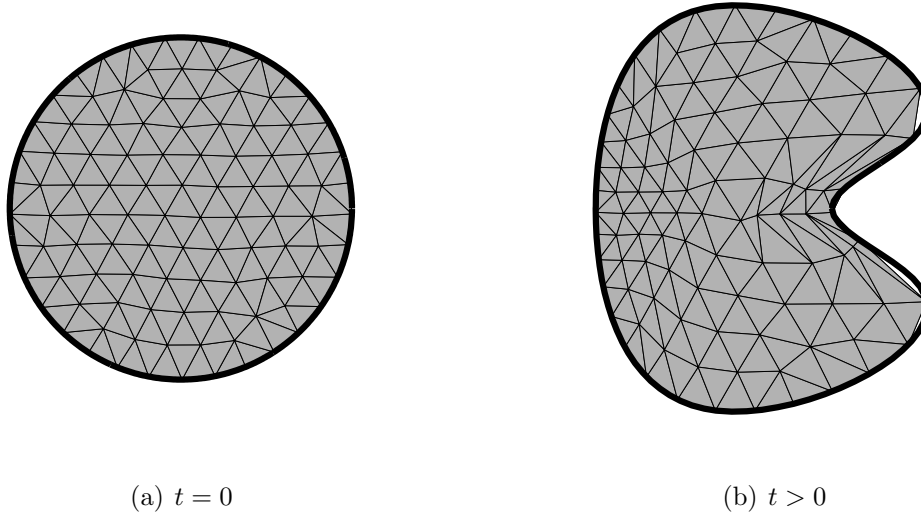


Figure 1: Schematic depiction of a deforming-mesh method. Without a careful choice of nodal motions, elements can suffer unwanted distortions under large deformations of the moving domain.²

paradigms schematically.

In this study, we eliminate these difficulties by employing a *universal mesh*: a stationary background mesh that adapts to the geometry of the immersed domain by adjusting a small number of mesh elements in the neighborhood of the moving boundary. An example is illustrated in Fig. 3. The resulting framework admits, in a general fashion, the construction of methods that are of arbitrarily high order of accuracy in space and time, without exhibiting the aforementioned drawbacks of deforming-mesh and fixed-mesh methods. This strategy was introduced for time-independent and quasi-steady problems in [4, 5]. Here we present its extension to time-dependent problems posed on moving domains with prescribed evolution. We relegate a discussion of problems with unprescribed boundaries to future work, since the treatment of unprescribed boundaries introduces its own set of challenges – approximation of the boundary, discretization of the boundary evolution equations, and error analysis on approximate domains – that may have the undesired effect of blurring the focus of the present study.

In the process of deriving our method, we present a unified, geometric framework that puts our method and existing deforming-mesh methods on a common footing suitable for analysis. The main idea is to recast the governing equations on a sequence of cylindrical spacetime slabs that span short intervals of time. The clarity brought about by this geometric viewpoint renders the analysis of numerical methods for moving-boundary problems more tractable, as it reduces the task to a standard analysis of fixed-domain problems with time-dependent PDE coefficients.

²Here, for purely illustrative purposes, we have employed a nodal mapping of the form $(r, \theta) \mapsto (f(\theta)r, \theta)$ in polar coordinates.

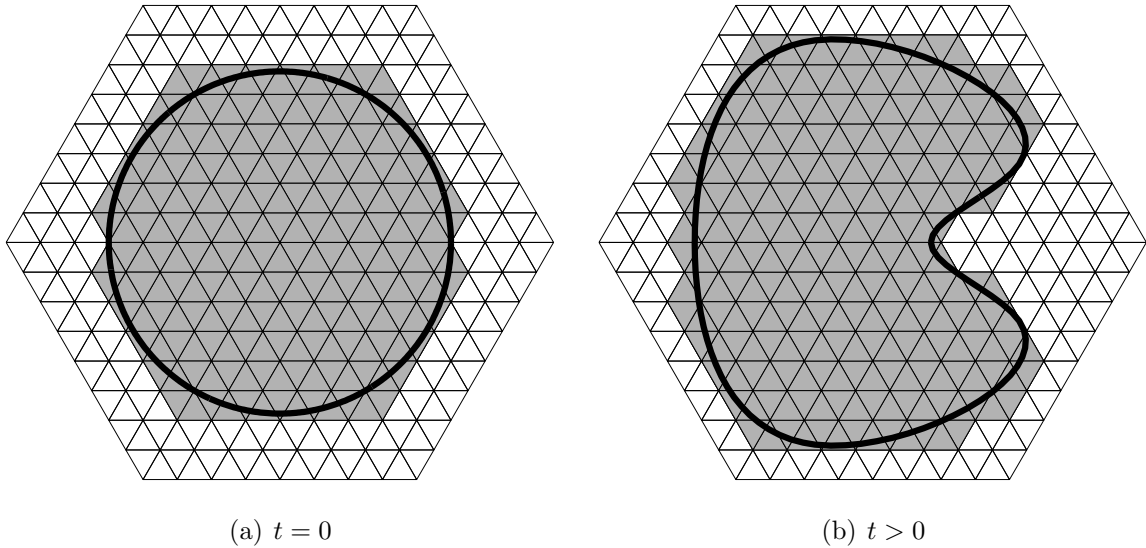


Figure 2: Schematic depiction of a fixed-mesh method. Such methods employ a fixed background mesh which does not conform to the immersed domain.

Organization. This paper is organized as follows. We begin in §2 by giving an informal overview of our method, and illustrating the ideas by formulating the method for a moving boundary problem in one spatial dimension. We formulate a two-dimensional model moving boundary problem on a predefined, curved spacetime domain in §3, and proceed to derive its equivalent reformulation on cylindrical spacetime slabs. In §4 we present, in an abstract manner, the general form of a finite-element discretization of the same moving boundary problem, as well as its reformulation on cylindrical spacetime slabs. This formalism will lead to a statement of the general form of a numerical method for moving boundary problems with prescribed boundary evolution that includes our method and conventional deforming-mesh methods as special cases. We finish §4 by summarizing an error estimate for methods of this form, referring the reader to our companion paper [6] for its proof. In §5, we present the key ingredient that distinguishes our proposed method from standard approaches: the use of a universal mesh. We specialize the aforementioned error estimate to this setting to deduce that the method’s convergence rate is suboptimal by half an order when the time step and mesh spacing scale proportionately. In §6 we demonstrate numerically our method’s convergence rate on a prescribed-boundary variant of a classic moving-boundary problem called the Stefan problem, which asks for the evolution of a solid-liquid interface during a melting process. Some concluding remarks are given in §7.

Previous work. In what follows, we review some of the existing numerical methods for moving-boundary problems, beginning with deforming-mesh methods and finishing with fixed-mesh methods.

Deforming-mesh methods have enjoyed widespread success in the scientific and engineering communities, where they are best known as Arbitrary Lagrangian Eulerian (ALE) methods. The appellation refers to the fact that in prescribing a motion of the mesh, a kinematic description of the physics is introduced that is neither Eulerian (in which the

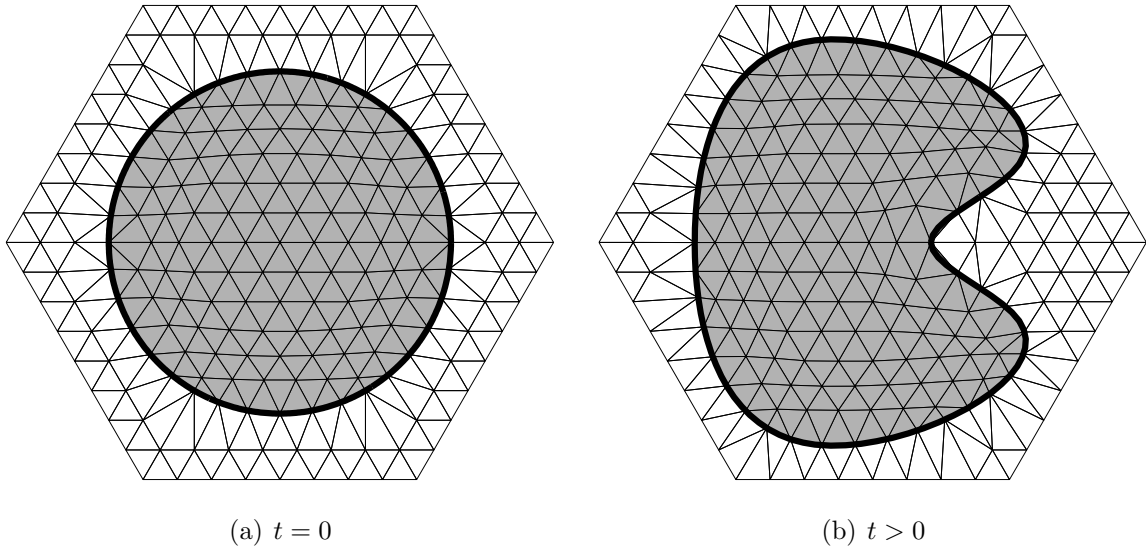


Figure 3: Schematic depiction of a universal mesh. By adapting the mesh to the immersed domain, one obtains a mesh that conforms to the domain exactly and is immune to large distortions of elements.

domain moves over a fixed mesh) nor Lagrangian (in which the domain does not move with respect to the mesh). The resulting formalism leads to governing equations that contain a term involving the velocity of the prescribed mesh motion that is otherwise absent in schemes on a fixed mesh [7, 8]. Early appearances of the ALE framework date back to the works of Hirt et. al. [9], Hughes et. al. [10], and Donea et. al. [11]. ALE methods have seen use in fluid-structure interaction [12, 13, 14, 15, 16, 17], solid mechanics [18, 19, 20, 21], thermodynamics [22, 23, 24, 25], and other applications.

Relative to methods for problems with fixed domains, less attention has been directed toward the development of ALE methods of high order of accuracy and the associated error analysis. Schemes of second-order in time are well-studied [15, 16, 13, 26, 27, 28, 29, 30], though the analysis of higher-order schemes has only recently been addressed by Bonito and co-authors [31, 32], who study the spatially continuous setting with discontinuous Galerkin temporal discretizations.

One of the key challenges that ALE methods face is the maintenance of a good-quality mesh during large deformations of the domain [33, 34]. Fig. 1 illustrates a case where, using an intentionally naive choice of nodal motions, a domain deformation can lead to triangles with poor aspect ratios. In more severe cases, element inversions can occur. Such distortions are detrimental both to the accuracy of the spatial discretization and to the conditioning of the discrete governing equations [35]. For this reason, it is common to use sophisticated mesh motion strategies that involve solving systems of equations (such as those of linear elasticity) for the positions of mesh nodes [36, 37, 38, 39].

A related class of methods are spacetime methods (e.g., [40]), where the spacetime domain swept out by the moving spatial domain is discretized with straight or curved elements. These methods resemble deforming-mesh methods in the sense that spatial slices of the spacetime mesh at fixed temporal nodes constitute a mesh of the moving domain at those times. Bonnerot and Jamet [41, 42] have used a spacetime framework to construct high-

order methods for the Stefan problem in one dimension. They require the use of curved elements along the moving boundary to achieve the desired temporal accuracy. Jamet [43] provides a generalization of these high-order methods to dimensions greater than one in the case that the boundary evolution is prescribed in advance. More recently, Rhebergen and Cockburn [44, 45] created hybridizable-discontinuous-Galerkin-based spacetime methods for advection-diffusion and incompressible flow problems with moving domains.

At the other extreme are fixed-mesh methods, which cover a sufficiently large domain with a mesh and evolve a numerical representation of the boundary, holding the background mesh fixed [46, 47, 48, 49]. A variety of techniques can be used to represent the boundary, including level sets [50, 2], marker particles [51], and splines [52]. Fixed mesh methods require that special care be taken in constructing the numerical partial differential operators in the neighborhood of the moving boundary, so as to avoid losses in accuracy arising from the disagreement between the moving boundary and element interfaces. Some authors [53, 54] propose adaptively refining the mesh in the neighborhood of the moving boundary to mitigate these losses. In the special case of a cartesian mesh, Gibou and Fedkiw [2] have developed a third-order method for the Stefan problem in two dimensions using extrapolation to allow finite-difference stencils to extend beyond the moving boundary.

The method presented in this paper classifies neither as a deforming-mesh method nor as a fixed-mesh method, though it shares attractive features from both categories. It exhibits the immunity to large mesh distortions enjoyed by fixed-mesh methods without sacrificing the geometric conformity offered by deforming-mesh methods. Despite its conceptual simplicity, the method has not been proposed in the literature. An idea similar to ours, dubbed a “fixed-mesh ALE” method, has recently been proposed by Baiges and Codina [55, 56], though there are several important differences. In particular, their method uses element splitting to define intermediate meshes during temporal integration, whereas our method leaves the connectivity of the mesh intact. Second, they advocate imposing boundary conditions approximately to improve efficiency; our method imposes boundary conditions exactly without extra computational effort. Finally, they focus only on low-order schemes with piecewise linear approximations to the domain deformation, while we derive schemes of arbitrarily high order.

2. Overview of the method

There are three main difficulties to overcome in constructing high-order methods for problems with moving domains: (a) Since the domain is changing in time, approximations of the domain of the appropriate order need to be constructed at all times at which the time-integration scheme is evaluated, (b) the approximation space over the evolving domain generally needs to evolve in time as well, resulting in a changing set of degrees of freedom, and (c) the approximation of time-derivatives of the solution near the evolving boundary needs to be carefully constructed, since solution values at a given spatial location may not be defined at all time instants within a time step.

Pulling back to a reference domain. A natural approach to sidestep these issues is to reformulate the problem as an evolution in a reference, fixed domain Ω^0 through a diffeomorphism $\varphi^t: \Omega^0 \rightarrow \Omega^t$ that maps it to the evolving domain Ω^t at each time t . If the solution sought is

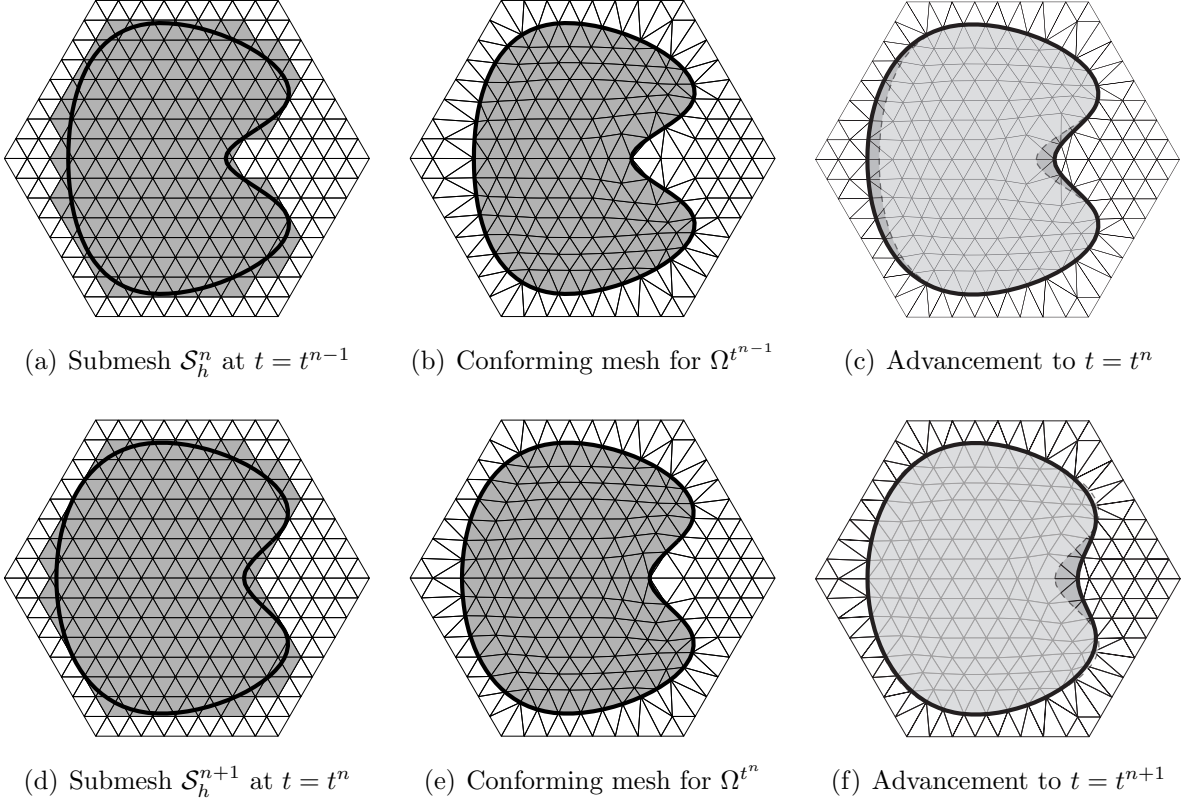


Figure 4: Sketch of how the reference domain is periodically redefined, and the mesh over it obtained. The triangles intersected by the domain in (a) are deformed through the universal mesh map to obtain a domain-matching discretization in (b). The evolution of the domain during $(t^{n-1}, t^n]$ is then described through a map φ^t defined over $\Omega^{t^{n-1}}$. The deformed mesh due to φ^{t^n} is then shown in (c), where the reference domain $\Omega^{t^{n-1}}$ is still depicted in light, transparent, gray. These steps are then repeated in (d), (e), and (f), for the interval $(t^n, t^{n+1}]$. The meshes in (c) and (e) both mesh Ω^{t^n} , but since the two differ near the domain boundary, a projection of the solution is needed to continue the integration in time.

$u(x, t)$, defined over the domain Ω^t at each time t , then this approach involves obtaining the partial differential equation that the function $U(X, t) = u(\varphi^t(X), t)$, defined over Ω^0 at all times, would satisfy. The obvious advantage of this perspective is that any of the standard numerical methods constructed for evolution problems on fixed domains can now be applied, and hence high-order methods can be easily formulated.

With this idea, the issues associated with discretizing an evolving domain are transformed into algorithmically constructing and computing the map φ^t . This is not too difficult when the changes in the domain are small, i.e., when φ^t is close to a rigid body motion for all times. However, it becomes challenging when φ^t induces large deformations of the domain. This is the typical problem of Arbitrary Lagrangian-Eulerian methods: how to deform the mesh, or alternatively, how to construct the ALE map (see Fig. 1). In terms of the map φ^t these same problems materialize as a loss of local or global injectivity.

A restatement of this same idea from a different perspective is to consider approximation spaces, such as a finite element spaces, that evolve with the domain. This is precisely what is obtained if each function in the approximation space over the reference configuration is

pushed forward by the map φ^t at each time t . For example, for finite element spaces, each shape function over Ω^t has the form $n_a(\varphi^t(X)) = N_a(X)$, where N_a is a shape function in the finite element space over Ω^0 . We take advantage of this equivalence throughout this manuscript.

Construction of maps. One of the central ideas we introduce here is one way to construct maps φ^t . To circumvent the problems that appear under large deformations, we periodically redefine the reference domain to be Ω^{t^n} , $n = 0, 1, \dots, N$, $t^n = n\tau$ for some $\tau > 0$, and accordingly $\varphi^t: \Omega^{t^n} \rightarrow \Omega^t$ for $t \in (t^n, t^{n+1}]$.

The combination of periodically redefining the reference configuration and constructing a mesh over it with the map proposed here is illustrated in Fig. 4, for a two-dimensional moving domain $\Omega^t \subset \mathbb{R}^2$. Upon choosing a fixed background triangulation \mathcal{T}_h of a domain $\mathcal{D} \subset \mathbb{R}^2$ that contains the domains Ω^t for all $t \in [0, T]$, $T = N\tau$, the method proceeds as follows: (a) At each temporal node t^{n-1} , a submesh \mathcal{S}_h^n of \mathcal{T}_h that approximates $\Omega^{t^{n-1}}$ (Fig. 4(a)) is identified; (b) The polygonal domain meshed by \mathcal{S}_h^n is deformed through the universal mesh map onto $\Omega^{t^{n-1}}$ (Fig. 4(b)); (c) The map φ^t for $t \in (t^{n-1}, t^n]$ is constructed as the identity everywhere except over the elements with one edge over the moving boundary. Over these elements φ^t is defined as an extension of the closest point projection of $\partial\Omega^{t^{n-1}}$ to $\partial\Omega^t$. Fig. 4(c) shows the mesh over Ω^{t^n} obtained as $\varphi^{t^n}(\Omega^{t^{n-1}})$. These three steps are repeated over $(t^n, t^{n+1}]$, as shown in Figs. 4(d), 4(e), and 4(f).

Discretization and time integration. As highlighted earlier, the introduction of the map φ^t enables the construction of approximations of any order within each interval $(t^{n-1}, t^n]$, and we elaborate on this next.

We denote the solution over $(t^{n-1}, t^n]$ with $U^{n-1}(X, t)$, which takes values over $\Omega^{t^{n-1}}$ at each time instant in this interval. To obtain appropriate spatial accuracy, notice that a finite element space of any order over $\Omega^{t^{n-1}}$ (Fig. 4(b)) can be defined in a standard way, by composing finite element functions over \mathcal{S}_h^n with the universal mesh map. The spatially discretized equations for U^{n-1} over this space form an ordinary system of differential equations whose unknowns are the degrees of freedom for U^{n-1} , and hence any standard, off-the-self integrator of any order can be adopted to approximate its solution.

The crucial role played by the universal mesh map is in full display here, since for smooth domains it provides an exact triangulation of $\Omega^{t^{n-1}}$. By ensuring that the mesh conforms exactly to the moving domain at all times, the method is free of geometric errors – errors that result from discrepancies between the exact domain and the computational approximation to the domain.

Projection. To continue the time integration from the interval $(t^{n-1}, t^n]$ to the interval $(t^n, t^{n+1}]$, an initial condition at t^n is needed, based on the solution computed in $(t^{n-1}, t^n]$. This initial condition is $U^n(x, t_+^n) = \lim_{t \searrow t^n} U^n(x, t) = U^{n-1}([\varphi^{t^n}]^{-1}(x), t)$, which is defined over Ω^{t^n} . In general, however, $U^n(x, t_+^n)$ does not belong to the discrete approximation space over Ω^{t^n} , so we project $U^n(x, t_+^n)$ onto it through a suitably defined projection operator; ideally an L^2 -projection, but numerical experiments with interpolation have rendered very good results as well.

The introduction of this projection N times would generally have the detrimental effect of reducing the order of convergence by one if the spacing τ between temporal nodes t^n is

proportional to the time step Δt adopted during integration over each interval $(t^{n-1}, t^n]$. Nevertheless, one of the highlights of the map φ^t we construct is that it differs from the identity in a region of thickness $O(h)$ from the domain boundary, where $h \sim \Delta t$ is the spatial mesh size. This feature makes the net reduction of the convergence rate due to the projection to be only of half an order (in the L^2 norm).

The implementation of this idea with finite element spaces is facilitated by regarding this method as a way to construct approximation spaces that evolve with the domain. This reduces the effect of the map φ^t to defining a “curved” mesh over Ω^t . By further interpolating the map φ^t with the finite element space, an isoparametric approximation of the domain is obtained. In this way, standard finite element procedures can be adopted to compute all needed quantities over either the exact or the isoparametric approximation of Ω^t . This curved mesh is constructed at each stage of the time-integration scheme.

Comparison with conventional ALE schemes. In light of the preceding paragraph, the reader may recognize that our method resembles a conventional ALE scheme with a peculiar mesh motion strategy and regular, systematic “remeshing.” In particular, the mesh motion defined by φ^t leaves all elements stationary except those with an edge on the moving boundary, and the “remeshing” entails the selection of a subtriangulation of a fixed background mesh and perturbing a few of its elements.

The peculiarity of the approach endows it with several unique features. Since the mesh motion is restricted to boundary elements, the lengths of the time intervals $(t^{n-1}, t^n]$ between “remeshing” (and hence the time step Δt adopted during time integration over those intervals) are restricted by the mesh spacing; see Section 5.5 for details. An advantage of this strategy is that it easily handles large domain deformations, and the nodal motions are independent and explicitly defined. However, for the reasons described earlier, the theoretical convergence rate of the method is suboptimal by half an order in the L^2 norm.

Remarks. Back to the difficulties highlighted at the beginning of this section, it should be evident by now that the basic idea we just outlined provides approximations of the domain of the proper order at all times, and that at no point does the difficulty of dealing with nodes that belong to Ω^t for only a fraction of the interval $(t^{n-1}, t^n]$ arise. The set of degrees of freedom in the approximation space does generally change because of the periodic redefinition of the reference configuration, a seemingly inevitable step for large enough deformations of the domain, but the introduction of the projection enables the continuation of the high-order integration in time with a minimal accuracy loss. We should also mention that a common difficulty for fixed-mesh methods, which is the imposition of Dirichlet or Neumann boundary conditions, is handled in a standard way with the approach in this manuscript.

In the following, we construct the method in one spatial dimension, to present some of the main ideas in a rigorous way, yet sidestepping the notational and algorithmic difficulties introduced by domain boundaries that are defined by curves instead of isolated points.

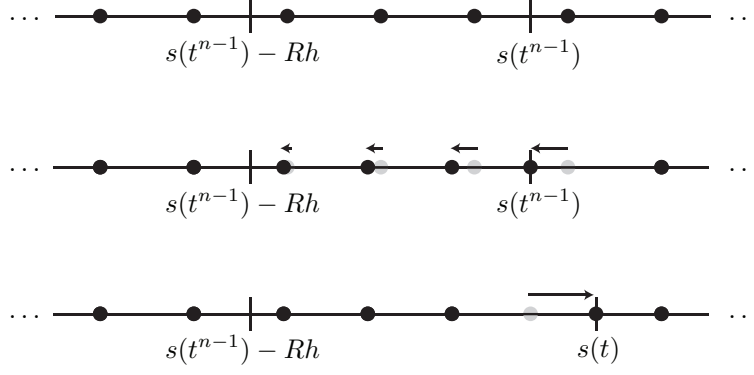


Figure 5: Illustration of the manner in which a one-dimensional universal mesh adapts to the immersed domain $(0, s(t))$ for $t \in (t^{n-1}, t^n]$. At $t = t_+^{n-1}$, the background mesh (top) is deformed by snapping the node that is closest to $s(t^{n-1})$ (among nodes outside the immersed domain) onto $s(t^{n-1})$ (middle). In the process, the nodes between $s(t^{n-1}) - Rh$ and $s(t^{n-1})$ are relaxed away from the boundary. At later times $t \in (t^{n-1}, t^n]$ (bottom), the snapped node tracks the position of the boundary, while all other nodes remain in the positions they adopted at $t = t_+^{n-1}$. Here, we used the map (2) with $R = 3$ and $\delta = 0.3$.

2.1. Construction of the method in one spatial dimension

Consider the moving boundary problem: Given a spacetime domain $\Omega = \{(x, t) \in \mathbb{R}^2 \mid 0 < x < s(t), 0 < t < T\}$, find $u: \Omega \rightarrow \mathbb{R}$ such that

$$0 = \frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2}, \quad (x, t) \in \Omega \quad (1a)$$

$$0 = u(0, t) = u(s(t), t), \quad 0 < t < T \quad (1b)$$

$$u^0(x) = u(x, 0), \quad 0 < x < s(0) \quad (1c)$$

where $s: [0, T] \rightarrow (0, 1)$ is a smooth, prescribed function of time, and $u^0: (0, s(0)) \rightarrow \mathbb{R}$ is the initial condition.

For such a problem, it suffices to adopt a grid $0 = X_0 < X_1 < \dots < X_M = 1$ of the unit interval as the universal mesh – a stationary background mesh that covers the domains $(0, s(t))$ for all times $0 \leq t \leq T$. We shall also employ a partition $0 = t^0 < t^1 < \dots < t^N = T$ of the time axis that is fine enough so that the change in $s(t)$ over a given interval $(t^{n-1}, t^n]$ never exceeds the minimum mesh spacing. That is,

$$\max_{t \in (t^{n-1}, t^n]} |s(t) - s(t^{n-1})| < \min_{0 \leq i \leq M} (X_i - X_{i-1}).$$

The universal mesh can be adapted to conform exactly to the domain $(0, s(t))$ at any time t by perturbing nodes in a small neighborhood of $s(t)$. A simple prescription for $t \in (t^{n-1}, t^n]$ is, for each i ,

$$x_i(t) = \begin{cases} X_i - \delta h \left(1 - \frac{s(t^{n-1}) - X_i}{Rh}\right) & \text{if } s(t^{n-1}) - Rh \leq X_i < s(t^{n-1}) \\ s(t) & \text{if } X_{i-1} < s(t^{n-1}) \leq X_i \\ X_i & \text{otherwise} \end{cases} \quad (2)$$

where R is a small positive integer, δ is a small positive number, and $h = \max_{0 < i \leq M} (X_i - X_{i-1})$. See Fig. 5 for an illustration. In this case, $\varphi^t(X) = \sum_{i=0}^M x_i(t) M_i(X)$, where M_i is the standard P_1 finite element shape function for node i : it is affine over each element and satisfies $M_i(X_j) = \delta_{ij}$.

On this adapted mesh we may construct shape functions $n_a(x, t) = N_a((\varphi^t)^{-1}(x))$, where $N_a(X)$ are the shape functions over the universal mesh. The shape functions n_a are (for instance) piecewise polynomial in x on each interval $[x_{i-1}(t), x_i(t)]$ for any fixed t , and are continuous in $t \in (t^{n-1}, t^n)$ for each fixed x . For each $t \in (t^{n-1}, t^n)$, the shape functions n_a satisfy that

$$\frac{\partial n_a}{\partial t}(x, t) = -\frac{\partial n_a}{\partial x}(x, t) v_h(x, t), \quad (3)$$

where $v_h(\varphi^t(X), t) = \frac{\partial}{\partial t} \varphi^t(X)$ is the (spatial/Eulerian) velocity of the adapted mesh. For $x_{i-1}(t) < x < x_i(t)$,

$$v_h(x, t) = \dot{x}_i(t) \left(\frac{x - x_{i-1}(t)}{x_i(t) - x_{i-1}(t)} \right) + \dot{x}_{i-1}(t) \left(\frac{x_i(t) - x}{x_i(t) - x_{i-1}(t)} \right).$$

We then seek an approximate solution

$$u_h(x, t) = \sum_{a=1}^A \mathbf{u}_a(t) n_a(x, t)$$

lying in the space of functions

$$\mathcal{V}_h(t) = \text{span}\{n_a(\cdot, t) : n_a(x, t) = 0 \forall x > s(t)\}.$$

Here, $\mathbf{u}(t) = (\mathbf{u}_1(t), \mathbf{u}_2(t), \dots, \mathbf{u}_A(t))^T \in \mathbb{R}^A$ is a vector of time-dependent coefficients, which we allow to be discontinuous across the temporal nodes t^n . We denote

$$\mathbf{u}(t_+^n) = \lim_{t \searrow t^n} \mathbf{u}(t)$$

and similarly for other scalar- or vector-valued functions. To obtain an equation for u_h , we perform a standard Galerkin projection of (1a) onto the space of functions $\mathcal{V}_h(t)$, which leads to the following ordinary differential equation for \mathbf{u} at each $t \in (t^{n-1}, t^n]$,

$$\mathbf{M}(t) \dot{\mathbf{u}}(t) - \mathbf{B}(t) \mathbf{u}(t) + \mathbf{K}(t) \mathbf{u}(t) = 0. \quad (4)$$

Here $\mathbf{M}(t) \in \mathbb{R}^{A \times A}$ is a mass matrix, $\mathbf{K}(t) \in \mathbb{R}^{A \times A}$ is a stiffness matrix, and $\mathbf{B}(t) \in \mathbb{R}^{A \times A}$ is an advection matrix, constructed according to the following prescription. For a such that

$$n_a(\cdot, t) \in \mathcal{V}_h(t),$$

$$\begin{aligned}\mathbf{M}_{ab}(t) &= \int_0^1 n_b(x, t) n_a(x, t) dx \\ \mathbf{B}_{ab}(t) &= \int_0^1 v_h(x, t) \frac{\partial n_b}{\partial x}(x, t) n_a(x, t) dx \\ \mathbf{K}_{ab}(t) &= \int_0^1 \frac{\partial n_b}{\partial x}(x, t) \frac{\partial n_a}{\partial x}(x, t) dx,\end{aligned}$$

while for a such that $n_a(\cdot, t) \notin \mathcal{V}_h(t)$,

$$\begin{aligned}\mathbf{M}_{ab}(t) &= 0 \\ \mathbf{B}_{ab}(t) &= 0 \\ \mathbf{K}_{ab}(t) &= \delta_{ab}.\end{aligned}$$

These last values are set so that $u_h(x, t) = 0$ for $x > s(t)$, which follows from imposing (1b). The algorithm then proceeds as follows:

Algorithm 2.1 Time integration for a universal mesh in one dimension.

Require: Initial condition $u(x, 0) = u^0(x)$.

- 1: **for** $n = 1, 2, \dots, N$ **do**
- 2: Project the current numerical solution

$$u_h(x, t^{n-1}) = \sum_{a=1}^A \mathbf{u}_a(t^{n-1}) n_a(x, t^{n-1})$$

(or the initial condition $u(x, 0)$ if $n = 1$) onto $\mathcal{V}_h(t_+^{n-1})$ to obtain the vector of coefficients $\mathbf{u}(t_+^{n-1})$ in the expansion

$$u_h(x, t_+^{n-1}) = \sum_{a=1}^A \mathbf{u}_a(t_+^{n-1}) n_a(x, t_+^{n-1}).$$

- 3: Numerically integrate

$$\mathbf{M}(t)\dot{\mathbf{u}}(t) - \mathbf{B}(t)\mathbf{u}(t) + \mathbf{K}(t)\mathbf{u}(t) = 0$$

for $t \in (t^{n-1}, t^n]$ with the initial condition $\mathbf{u}(t_+^{n-1})$ and the constraints induced by (1b) to obtain $\mathbf{u}(t^n)$.

- 4: **end for**
 - 5: **return** $u_h(x, t^N)$
-

Several salient features of the method should be evident at this point:

- *The connectivity of the universal mesh never changes* during deformation – only the nodal positions change. As a consequence, the sizes and sparsity structures of various

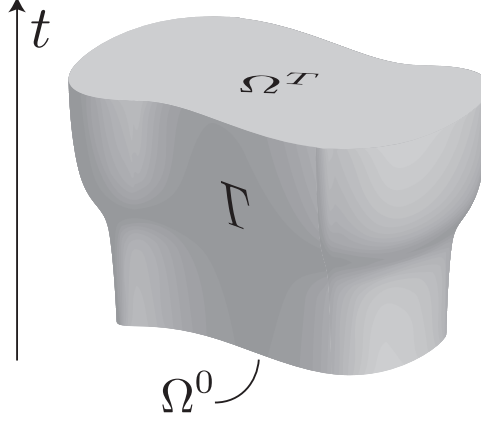


Figure 6: Spacetime domain Ω .

discrete quantities (the solution vector \mathbf{u} , the mass matrix \mathbf{M} , the stiffness matrix \mathbf{K} , and the advection matrix \mathbf{B}) can be held fixed, even though differing subsets of degrees of freedom may participate in the discrete equations at any interval $(t^{n-1}, t^n]$. One merely needs to impose “homogeneous Dirichlet boundary conditions” on the solution at nonparticipating degrees of freedom.

- Large deformations of the domain pose no threat to the quality of the deformed mesh, provided $\max_{1 \leq n \leq N} (t^n - t^{n-1})$ is sufficiently small and the domain evolution is sufficiently regular.
- In two dimensions, the nodal motions are independent and explicitly defined, rendering the mesh motion strategy low-cost and easily parallelizable. See Section 5 for details.

3. A Model Moving Boundary Problem

3.1. The Continuous Problem

Consider a moving boundary problem on a bounded spacetime domain $\Omega \subset \mathbb{R}^2 \times [0, T]$, as in Fig. 6. For each $t \in [0, T]$, denote by $\Omega^t \subset \mathbb{R}^2$ the spatial component of the spacetime slice $\Omega \cap (\mathbb{R}^2 \times \{t\})$, and denote by Γ^t the boundary of Ω^t . Finally, let $\Gamma = \bigcup_{0 < t < T} (\Gamma^t \times \{t\})$ denote the lateral boundary of the spacetime domain Ω . We assume that Ω^t is open in \mathbb{R}^2 for each t . As a regularity requirement, we assume that for every $t \in [0, T]$, the set Γ^t can be expressed as the image of an embedding $c(\cdot, t)$ of the unit circle S^1 into \mathbb{R}^2 , where $c \in C^2(S^1 \times (0, T), \mathbb{R}^2)$.

Now consider the following abstract moving boundary problem: Given $f: \Omega \rightarrow \mathbb{R}$ and $u^0: \Omega^0 \rightarrow \mathbb{R}$, find $u: \Omega \rightarrow \mathbb{R}$ satisfying

$$\frac{\partial u}{\partial t} + a(u) = f \text{ in } \Omega \tag{5a}$$

$$u = 0 \quad \text{on } \Gamma \tag{5b}$$

$$u = u^0 \quad \text{on } \Omega^0, \tag{5c}$$

where a is a partial differential operator of the form

$$a(u) = -\nabla_x \cdot (k_1 \nabla_x u) + k_2 \cdot \nabla_x u + k_3 u$$

with coefficients $k_1(x, t) \in \mathbb{R}^{2 \times 2}$, $k_2(x, t) \in \mathbb{R}^2$, and $k_3(x, t) \in \mathbb{R}$ for every $(x, t) \in \Omega$. We assume that k_1 is uniformly positive definite. That is, there exists $C > 0$ such that $v \cdot k_1(x, t)v \geq Cv \cdot v$ for every $v \in \mathbb{R}^2$ and every $(x, t) \in \Omega$.

It is known [57, Theorem 7.17] that if $k_1 \in L^\infty(\Omega)^{2 \times 2}$, $k_2 \in L^\infty(\Omega)^2$, $k_3 \in L^\infty(\Omega)$, the components of k_1 are Lipschitz in spacetime, $f \in L^p(\Omega)$, and $u^0 \in W^{2,p}(\Omega^0)$ with $1 < p < \infty$, then the problem (5) has a unique solution u with $u(\cdot, t) \in W^{2,p}(\Omega^t)$ and $\frac{\partial u}{\partial t}(\cdot, t) \in L^p(\Omega^t)$ for every $0 \leq t \leq T$. Here, $W^{s,p}$ denotes the Sobolev space of differentiability $s \geq 0$ and integrability $1 \leq p \leq \infty$, and $L^p = W^{0,p}$ denotes the Lebesgue space of integrability $1 \leq p \leq \infty$. Later, we shall also denote $H^s = W^{s,2}$, and we write $H_0^1(\Omega^t)$ for the space of functions in $H^1(\Omega^t)$ with vanishing trace. We denote the norm on $W^{s,p}(\Omega^t)$ by $\|\cdot\|_{s,p,\Omega^t}$ and the associated semi-norm by $|\cdot|_{s,p,\Omega^t}$.

3.2. Equivalent Formulation of the Continuous Problem

In the following, we derive an equivalent formulation of the moving-boundary problem (5) that is well-suited for numerical discretization. For reasons that will soon be made clearer, we restrict our attention to a temporal subinterval $(t^{n-1}, t^n] \subset [0, T]$ for the remainder of this section.

Weak formulation. A weak formulation of (5) reads: Find $u(\cdot, t) \in \mathcal{V}(\Omega^t) := H_0^1(\Omega^t)$ such that

$$m^t(\dot{u}, w) + a^t(u, w) = m^t(f, w) \quad \forall w \in \mathcal{V}(\Omega^t) \quad (6)$$

for every $t \in (t^{n-1}, t^n]$, where the time-dependent bilinear forms m^t and a^t are given by

$$\begin{aligned} m^t(u, w) &= \int_{\Omega^t} uw \, dx \\ a^t(u, w) &= \int_{\Omega^t} \nabla_x w \cdot k_1 \nabla_x u + (k_2 \cdot \nabla_x u)w + k_3 uw \, dx. \end{aligned}$$

Here and throughout this paper, the dot notation denotes differentiation with respect to time while holding the remaining arguments to the function fixed.

Pulling back to a cylindrical domain. Given any sufficiently smooth family of bijections $\{\varphi^{n,t} : \Omega^{t^{n-1}} \rightarrow \Omega^t \mid t \in (t^{n-1}, t^n]\}$, equation (6) may be recast on the cylindrical spacetime domain $\Omega^{t^{n-1}} \times (t^{n-1}, t^n]$, since, by a change of variables, (6) is equivalent to the statement

$$M^t(\dot{U}, W) - B^t(U, W) + A^t(U, W) = M^t(F, W) \quad \forall W \in (\varphi^{n,t})^* \mathcal{V}(\Omega^t) \quad (7)$$

for every $t \in (t^{n-1}, t^n]$, where

$$(\varphi^{n,t})^* \mathcal{V}(\Omega^t) = \left\{ W : \Omega^{t^{n-1}} \rightarrow \mathbb{R} \mid W = w \circ \varphi^{n,t} \text{ for some } w \in \mathcal{V}(\Omega^t) \right\}$$

is the space of functions in $\mathcal{V}(\Omega^t)$ pulled back to $\Omega^{t^{n-1}}$ by $\varphi^{n,t}$,

$$\dot{U}(X, t) = \left. \frac{\partial}{\partial t} \right|_X U(X, t),$$

and

$$\begin{aligned} M^t(U, W) &= \int_{\Omega^{t^{n-1}}} UW |\nabla_X \varphi^{n,t}| dX \\ B^t(U, W) &= \int_{\Omega^{t^{n-1}}} ((\nabla_X \varphi^{n,t})^{-\dagger} \nabla_X U \cdot V^{n,t}) W |\nabla_X \varphi^{n,t}| dX \\ A^t(U, W) &= \int_{\Omega^{t^{n-1}}} \left[((\nabla_X \varphi^{n,t})^{-\dagger} \nabla_X W) \cdot K_1 ((\nabla_X \varphi^{n,t})^{-\dagger} \nabla_X U) \right. \\ &\quad \left. + ((\nabla_X \varphi^{n,t})^{-\dagger} \nabla_X U \cdot K_2) W + K_3 U W \right] |\nabla_X \varphi^{n,t}| dX, \end{aligned}$$

with $|\nabla_X \varphi^{n,t}|$ denoting the absolute value of the Jacobian determinant of $\varphi^{n,t}$ and $(\nabla_X \varphi^{n,t})^{-\dagger}$ denoting the inverse adjoint of $\nabla_X \varphi^{n,t}$. Here, $K_i = k_i \circ \varphi^{n,t}$, $i = 1, 2, 3$ and $F = f \circ \varphi^{n,t}$ are the Lagrangian counterparts of k_1, k_2, k_3 , and f , and

$$V^{n,t}(X) := \dot{\varphi}^{n,t}(X) = \left. \frac{\partial}{\partial t} \right|_X \varphi^{n,t}(X)$$

is the *material* or *Lagrangian velocity*.

The validity of the preceding change of variables will hold if, for instance,

$$t \mapsto \varphi^{n,t} \in C^1 \left((t^{n-1}, t^n], W^{1,\infty}(\Omega^{t^{n-1}})^2 \right), \quad (8)$$

and $(\varphi^{n,t})^{-1} \in W^{1,\infty}(\Omega^t)^2$ for $t \in (t^{n-1}, t^n]$. Note that under these assumptions, $(\varphi^{n,t})^* \mathcal{V}(\Omega^t) = \mathcal{V}(\Omega^{t^{n-1}}) = H_0^1(\Omega^{t^{n-1}})$.

The presence of the term $B^t(U, W)$ in (7) arises from the identity

$$\frac{\partial U}{\partial t}(X, t) = \frac{\partial u}{\partial t}(\varphi^{n,t}(X), t) + \nabla_x u(\varphi^{n,t}(X), t) \cdot v^{n,t}(\varphi^{n,t}(X)), \quad (9)$$

which relates the partial time derivative of u to the *material time derivative*

$$\frac{Du}{Dt}(\varphi^{n,t}(X), t) := \frac{\partial U}{\partial t}(X, t)$$

of u via a term involving the *spatial* or *Eulerian velocity*

$$v^{n,t}(\varphi^{n,t}(X)) = V^{n,t}(X).$$

Upon discretization, the term $B^t(U, W)$ corresponds precisely to the term $\mathbf{B}(t)\mathbf{u}(t)$ that the reader encountered earlier in (4).

“Hybrid” Eulerian formulation. A third equivalent statement of (6) and (7) is obtained by acknowledging that, by (9),

$$\frac{\partial u}{\partial t}(x, t) = \frac{Du}{Dt}(x, t) - \nabla_x u(x, t) \cdot v^{n,t}(x). \quad (10)$$

It then follows that (6) is equivalent to

$$m^t \left(\frac{Du}{Dt}, w \right) - b^t(u, w) + a^t(u, w) = m^t(f, w) \quad \forall w \in \mathcal{V}(\Omega^t) \quad (11)$$

for every $t \in (t^{n-1}, t^n]$, where the time-dependent bilinear form b^t is given by

$$b^t(u, w) = \int_{\Omega^t} \nabla_x u \cdot v^{n,t} w \, dx. \quad (12)$$

So, u satisfies (6) if and only if it satisfies (11) and if and only if U satisfies (7). The advantage of this formulation is that it involves simpler expressions for the bilinear forms than those in (7), and these simpler expressions will be convenient for the numerical implementation later. Notice as well that the material time derivative on $\partial\Omega^t$ is now a directional derivative in a direction tangential to the spacetime boundary $\partial\Omega$, in contrast to \dot{u} , which can only be defined as a one-side derivative therein.

4. Discretization

4.1. Spatial Discretization on Short Time Intervals

At this point it is instructive to derive, in a systematic manner, the general form of a finite element spatial discretization of (5) obtained via Galerkin projection. We begin by spatially discretizing the weak formulation (6) and proceed by pulling the semidiscrete equations back to a cylindrical spacetime domain, and by obtaining the “hybrid” Eulerian formulation of the same semidiscrete equations. The utility of these three formulations will be evident towards the end of this section.

Galerkin formulation. A Galerkin projection of (6) requires choosing a finite-dimensional subspace $\mathcal{V}_h(\Omega^t) \subset \mathcal{V}(\Omega^t)$ at each time t and finding $u_h(t) \in \mathcal{V}_h(\Omega^t)$ such that

$$m^t(\dot{u}_h, w_h) + a^t(u_h, w_h) = m^t(f, w_h) \quad \forall w_h \in \mathcal{V}_h(\Omega^t) \quad (13)$$

for every $t \in (t^{n-1}, t^n]$. For concreteness, let us construct such a family of finite element spaces by fixing a reference triangulation \mathcal{S}_h^n of a polygonal domain $\mathcal{D}(\mathcal{S}_h^n) \subset \mathbb{R}^2$ and constructing a family of continuous, bijective maps

$$\Phi_h^{n,t} : \mathcal{D}(\mathcal{S}_h^n) \rightarrow \Omega^t$$

that are differentiable in time and are affine on each triangle $K \in \mathcal{S}_h^n$, except perhaps near the boundary, see Fig. 7. In informal language, the image of $\Phi_h^{n,t}$ provides a moving mesh

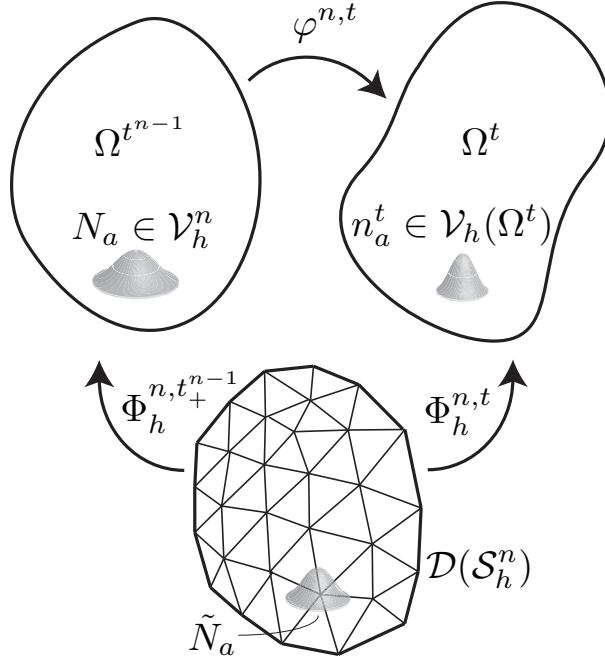


Figure 7: For each $t \in (t^{n-1}, t^n]$, the map $\Phi_h^{n,t}$ provides a bijection from a fixed reference triangulation \mathcal{S}_h^n of a polygonal domain $\mathcal{D}(\mathcal{S}_h^n)$ to the moving domain Ω^t . Depicted pictorially is a shape function \tilde{N}_a on the reference triangulation and its pushforward to $\Omega^{t^{n-1}}$ and Ω^t , denoted N_a and n_a^t , respectively.

that triangulates Ω^t for each $t \in (t^{n-1}, t^n]$. Then, with $\{\tilde{N}_a\}_a$ denoting shape functions on the reference triangulation, we may set

$$\mathcal{V}_h(\Omega^t) = \text{span}\{n_a^t\}_a \quad (14)$$

with

$$n_a^t = \tilde{N}_a \circ (\Phi_h^{n,t})^{-1}$$

for each $t \in (t^{n-1}, t^n]$.

Pulling back to a cylindrical domain. We may pull back the semidiscrete equations (13) to the cylindrical spacetime domain $\Omega^{t^{n-1}} \times (t^{n-1}, t^n]$ with the aid of the bijections

$$\varphi^{n,t} := \Phi_h^{n,t} \circ (\Phi_h^{n,t^{n-1}})^{-1}. \quad (15)$$

The resulting equivalent semidiscrete equation reads

$$M^t(\dot{U}_h, W_h) - B^t(U_h, W_h) + A^t(U_h, W_h) = M^t(F, W_h) \quad \forall W_h \in (\varphi^{n,t})^* \mathcal{V}_h(\Omega^t) \quad (16)$$

for every $t \in (t^{n-1}, t^n]$.

“Hybrid” Eulerian formulation. Similarly, the discrete “hybrid” Eulerian formulation follows by taking advantage of (9) to replace \dot{u}_h in (13), to get

$$m^t \left(\frac{Du_h}{Dt}, w_h \right) - b^t(u_h, w_h) + a^t(u_h, w_h) = m^t(f, w_h) \quad \forall w_h \in \mathcal{V}_h(\Omega^t) \quad (17)$$

for every $t \in (t^{n-1}, t^n]$.

Remark. We note that (13), (16), and (17) do not define three different methods; they are three ways of writing precisely the same one. That is, u_h satisfies (13) if and only if it satisfies (17) and if and only if $U_h(t) = (\varphi^{n,t})^* u_h(t)$ satisfies (16).

Finite element spaces. Notice that (16) is a discretization of (7) with a particular choice of a finite element subspace of $\mathcal{V}(\Omega^{t^{n-1}})$, namely $(\varphi^{n,t})^* \mathcal{V}_h(\Omega^t)$. The shape functions for this space are given by

$$\begin{aligned} N_a &= n_a^t \circ \varphi^{n,t} \\ &= \tilde{N}_a \circ (\Phi_h^{n,t^{n-1}})^{-1}, \end{aligned}$$

which are time-independent.

As a consequence, the material time derivative of functions in $\mathcal{V}_h(\Omega^t)$ takes a particularly simple form. Let

$$u_h(\varphi^{n,t}(X), t) = \sum_a u_a(t) n_a^t(\varphi^{n,t}(X)) = \sum_a u_a(t) N_a^t(X) = U_h(X, t).$$

Then

$$\frac{Du_h}{Dt}(\varphi^{n,t}(X), t) = \frac{\partial U_h}{\partial t}(X, t) = \sum_a \dot{u}_a(t) N_a^t(X) = \sum_a \dot{u}_a(t) n_a^t(\varphi^{n,t}(X)), \quad (18)$$

since the shape functions $\{N_a\}_a$ do not depend on time.

Since the map (15) depends upon h , we make that dependence explicit by appending a subscript h to $\varphi^{n,t}$ and all derived quantities ($v^{n,t}$, $V^{n,t}$, M^t , A^t , and B^t) in the remainder of this text.

Summary. In summary, we have shown that if the semidiscrete equation (13) is pulled back to the reference domain $\Omega^{t^{n-1}}$ through the use of a map

$$\varphi_h^{n,t} = \Phi_h^{n,t} \circ (\Phi_h^{n,t^{n-1}})^{-1},$$

then the resulting semidiscrete equation (16) involves a finite element space that does not change with time. We may label that space \mathcal{V}_h^n and write

$$M_h^t(\dot{U}_h, W_h) - B_h^t(U_h, W_h) + A_h^t(U_h, W_h) = M_h^t(F, W_h) \quad \forall W_h \in \mathcal{V}_h^n \quad (19)$$

for every $t \in (t^{n-1}, t^n]$. The shape functions for \mathcal{V}_h^n are simply shape functions on the reference triangulation \mathcal{S}_h^n pushed forward to $\Omega^{t^{n-1}}$:

$$N_a = \tilde{N}_a \circ (\Phi_h^{n,t^{n-1}})^{-1}.$$

The utility of the above formulation is transparent. Upon expanding U_h as a linear combination of shape functions, the system (19) is a system of ordinary differential equations for the coefficients of the expansion. This is also evident from the “hybrid” Eulerian formulation (17) upon replacing the material time derivative by (18). To this system of ODEs we may apply a time integrator of choice to advance from time t^{n-1} to time t^n .

4.2. Integration over Long Time Intervals

In the preceding sections, we elected to restrict our attention to a temporal subinterval $(t^{n-1}, t^n] \subset [0, T]$ and construct finite element subspaces of $\mathcal{V}(\Omega^t)$, $t \in (t^{n-1}, t^n]$, using a smoothly varying triangulation of Ω^t given by the image of $\Phi_h^{n,t}$, $t \in (t^{n-1}, t^n]$. This decision allows for the use of different reference triangulations \mathcal{S}_h^n on different temporal subintervals, simplifying the task of maintaining a nondegenerate triangulation of a domain undergoing large deformations.

To complete the picture and construct an algorithm for integration over the interval $[0, T]$ of interest, we choose a partition $0 = t^0 < t^1 < \dots < t^N = T$ and make use of one last ingredient: a linear projector p_h^n onto \mathcal{V}_h^n for each n . For the definition of the algorithm, we require that the domain of definition of p_h^n contains at least the space \mathcal{U}_h^n given by

$$\mathcal{U}_h^n = \begin{cases} \mathcal{V}(\Omega^0) & \text{if } n = 1 \\ (\varphi_h^{n-1, t^{n-1}})_* \mathcal{V}_h^{n-1} + \mathcal{V}_h^n & \text{if } 1 < n \leq N, \end{cases}$$

where

$$(\varphi_h^{n-1, t^{n-1}})_* \mathcal{V}_h^{n-1} = \left\{ w : \Omega^{t^{n-1}} \rightarrow \mathbb{R} \mid w \circ \varphi_h^{n-1, t^{n-1}} \in \mathcal{V}_h^{n-1} \right\}$$

is the space of functions in \mathcal{V}_h^{n-1} pushed forward to $\Omega^{t^{n-1}}$ by $\varphi_h^{n-1, t^{n-1}}$. We assume the projector is surjective for each n ; equivalently, $p_h^n|_{\mathcal{V}_h^n} = \text{identity}$ for each n .

Some examples of projectors are the orthogonal projector p_{h, L^2}^n onto \mathcal{V}_h^n with respect to the L^2 -inner product, the orthogonal projector p_{h, H^1}^n onto \mathcal{V}_h^n with respect to the H^1 -inner product, and the nodal interpolant i_h^n onto \mathcal{V}_h^n ; see [58, Chapter 1] for details. The appropriate projector depends on the problem being approximated and the choice of temporal nodes t^n , triangulations \mathcal{S}_h^n , and maps $\Phi_h^{n,t}$. As we shall mention, p_{h, L^2}^n is the projector best suited for use with the choices detailed in Section 5.

With such a family of projectors at hand, a method for integration over the full time interval $[0, T]$ is then summarized in Algorithm 4.1.

Relationship to ALE. Let us emphasize that Algorithm 4.1 has been formulated with enough generality that it encompasses not only the method specific to this paper involving universal meshes (which is detailed in Section 5) but also conventional ALE schemes. In the case of an ALE scheme, the reference triangulation \mathcal{S}_h^n is a triangulation of $\Omega^{t^{n-1}}$, the map $\varphi_h^{n,t}$ corresponds to a mesh motion derived from, e.g., solutions to the equations of linear elasticity, and the temporal nodes t^n correspond to times at which remeshing is performed. In the case of the method specific to this paper, we shall see in Section 5 that the reference triangulation \mathcal{S}_h^n is a subtriangulation of a fixed background mesh, the map $\varphi_h^{n,t}$ induces deformations of triangles on the boundary of \mathcal{S}_h^n while leaving the remaining triangles fixed, and the temporal nodes t^n are spaced closely enough so that these deformations of boundary triangles remain well-behaved.

4.3. Example: a Runge-Kutta Time-Integrator

We next exemplify how a time integrator of any given order can be incorporated into step 5 of the algorithm. In this case we consider an s -stage Singly Diagonally Implicit Runge-Kutta (SDIRK) method of order $\leq s$ as the time integrator [59, 60]. Such an integrator

Algorithm 4.1 General form of a time integrator for moving-boundary problems with a finite element discretization in space.

Require: Initial condition $u^0 \in \mathcal{V}(\Omega^0)$.

- 1: **for** $n = 1, 2, \dots, N$ **do**
- 2: Choose a reference triangulation \mathcal{S}_h^n and a family of maps $\Phi_h^{n,t} : \mathcal{D}(\mathcal{S}_h^n) \rightarrow \Omega^t$, $t \in (t^{n-1}, t^n]$.
- 3: Generate a finite-dimensional subspace \mathcal{V}_h^n of the continuous solution space $\mathcal{V}(\Omega^{t^{n-1}})$ using shape functions on \mathcal{S}_h^n composed with $(\Phi_h^{n,t^{n-1}})^{-1}$.
- 4: Project the current numerical solution (or the initial condition if $n = 1$) onto \mathcal{V}_h^n by setting

$$U_h(\cdot, t_+^{n-1}) = p_h^n u_h(\cdot, t^{n-1}),$$

where $u_h(\cdot, t^0) = u^0$ or, for $n > 1$,

$$u_h(x, t^{n-1}) = U_h((\varphi_h^{n-1, t^{n-1}})^{-1}(x), t^{n-1})$$

is the pushforward of $U_h(\cdot, t^{n-1}) \in \mathcal{V}_h^{n-1} \subset \mathcal{V}(\Omega^{t^{n-2}})$ to $\Omega^{t^{n-1}}$.

- 5: Numerically integrate (19) over $(t^{n-1}, t^n]$ with the projected initial condition $U_h(\cdot, t_+^{n-1})$.
 - 6: **end for**
 - 7: **return** $u_h(\cdot, t^N)$
-

requires solving a sequence of s systems of equations

$$M_h^{t_i}(U_i, W) = M_h^{t_i} \left(\sum_{j=0}^{i-1} \beta_{ij} U_j, W \right) + \gamma \Delta t G_h^{t_i}(U_i, F(t_i); W) \quad \forall W \in \mathcal{V}_h^n \quad (20)$$

for $U_i \in \mathcal{V}_h^n$, $i = 1, 2, \dots, s$, where $U_0 = U_h(\cdot, t_0)$, $t_0 \in (t^{n-1}, t^n]$, $t_i = \sum_{j=0}^{i-1} \beta_{ij} t_j + \gamma \Delta t$ for $0 < i \leq s$, and

$$G_h^t(U, F; W) = M_h^t(F, W) - A_h^t(U, W) + B_h^t(U, W).$$

The time- Δt advancement of U_0 is then given by U_s . The coefficients $\gamma > 0$ and $\beta_{ij} \in \mathbb{R}$, $i = 1, 2, \dots, s$, $j = 0, 1, \dots, i-1$, for various SDIRK methods are tabulated in Appendix A, Tables A.3-A.5. Pragmatically, implementing an SDIRK method amounts to computing s “backward-Euler” steps, with the initial condition at the i^{th} stage given by a linear combination of the solutions at the previous stages.

4.4. Overview of Error Estimates

In our companion paper [6], we derive error estimates in the L^2 -norm for the aforementioned method for the problem in §3.1. Here, we give an overview of the estimates for the case in which the finite element spaces \mathcal{V}_h^n consist of continuous functions made of element-wise polynomials of degree $r-1$, where $r > 1$ is an integer. We begin by introducing some notation.

Notation. Let $u^n \in \mathcal{V}(\Omega^{t^n})$ denote the value of the exact solution u at $t = t^n$, i.e. $u^n = u(\cdot, t^n)$, and let $u_h^{\Delta t, n} \in \mathcal{V}_h(\Omega^{t^n})$ denote the value of the fully discrete solution at $t = t^n$. Finally, let $v_h : \Omega \rightarrow \mathbb{R}^2$ denote the vector field on the spacetime domain Ω whose restriction to each temporal slice is $v_h^{n,t}$, i.e. $v_h(\cdot, t) = v_h^{n,t}$ for $t \in (t^{n-1}, t^n]$.

From this point forward, the parameter h denotes the maximum diameter of a triangle belonging to $\{K \in \mathcal{S}_h^n \mid 1 \leq n \leq N\}$. Additionally, let Δt be the maximum time step adopted while time-integrating over the interval $[0, T]$, namely, the maximum time step employed by the time-integrator in Line 5 of Algorithm 4.1 among all intervals $(t^{n-1}, t^n]$. We remind the reader that the temporal nodes t^n demarcate changes in the reference triangulation \mathcal{S}_h^n ; hence, the time step adopted during integration over $(t^{n-1}, t^n]$ is less than or equal to $t^n - t^{n-1}$ for every n . We assume that the time integrator employed during these intervals is stable and has a global truncation error of order $q \geq 1$ in the time step Δt .

In Line 4 of of Algorithm 4.1, the numerical solution is transferred, via a projection p_h^n , between two finite element spaces associated with differing triangulations of $\Omega^{t^{n-1}}$. We denote by $\mathcal{R}_h^n \subseteq \Omega^{t^{n-1}}$ the region over which the two triangulations differ, and by $|\mathcal{R}_h^n|$ its (Lebesgue) measure. We assume the projector is stable in the sense that there exists a constant C_p independent of h and n such that

$$\|p_h^n U\|_{0,2,\Omega^{t^{n-1}}} \leq C_p \|U\|_{0,2,\Omega^{t^{n-1}}}$$

for all $U \in \mathcal{U}_h^n$.

General error estimate. It is proven in [6] that if the assumptions above are satisfied, the triangulations \mathcal{S}_h^n are quasi-uniform, and the exact solution u and the maps $\Phi_h^{n,t}$ are sufficiently regular, then an error estimate of the following form holds with constants $C_1(u, v_h, T)$, $C_2(u, v_h, T)$ and $C_3(u, T)$:

$$\|u_h^{\Delta t, N} - u^N\|_{0,2,\Omega^T} \leq C_p^N \left(C_1(u, v_h, T) h^r + C_2(u, v_h, T) \Delta t^q + C_3(u, T) h^r \ell_{h,r} \sum_{n=1}^N |\mathcal{R}_h^n|^{1/2} \right) \quad (21)$$

where

$$\ell_{h,r} = \begin{cases} \log(h^{-1}) & \text{if } r = 2 \\ 1 & \text{if } r > 2 \end{cases} \quad (22)$$

and $C_1(u, v_h, T), C_2(u, v_h, T) \geq C(u, T)$ for some constant $C(u, T) > 0$.

The content of this estimate is easily understood. The error committed by the method consists of three terms, amplified by the N^{th} power of the projector's stability constant: an error due to spatial discretization of order at best h^r (first term), an error due to temporal discretization of order at best Δt^q (second term), and an error introduced by projecting between differing triangulations of the same domain Ω^{t^n} at each temporal node t^n (third term). The coefficients of the first two terms depend implicitly on h through the choice of the domain velocity v_h . The precise scaling of these coefficients with respect to h depends upon the chosen mesh motion strategy and is, of course, no better than $O(1)$ in h . We particularize this estimate for the mesh motion strategy proposed here in §5.5.

Discussion. Error estimates that are specific to two categories of methods are immediately apparent from the general estimate (21). The first category consists of classical ALE schemes with occasional remeshing – that is, N is independent of h and Δt . For methods of this type, the amplifier C_p^N is of order unity (regardless of the choice of the projector), the summation of the mesh discrepancy volumes is of order unity, and the coefficients $C_1(u, v_h, T)$ and $C_2(u, v_h, T)$ can be bounded independently of h for sufficiently regular mesh motion strategies. The resulting error estimate reads

$$\|u_h^{\Delta t, N} - u^N\|_{0,2,\Omega^T} \leq C(u, T)(h^r + \Delta t^q + \ell_{h,r} h^r)$$

for a constant $C(u, T)$ (it is straightforward to sharpen this estimate to $\|u_h^{\Delta t, N} - u^N\|_{0,2,\Omega^T} \leq C(u, T)(h^r + \Delta t^q)$).

At the other extreme are methods for which the reference triangulation \mathcal{S}_h^n is updated more frequently, e.g., at intervals proportional to Δt . This strategy is, in fact, the one adopted in the method proposed in Section 5. For methods of this type, the proportionality between N and Δt^{-1} mandates the use of a projector with stability constant $C_p = 1$ (such as the L^2 -projector p_{h,L^2}^n). However, the short time intervals between updates of the reference triangulation allow for the use of simple mesh motion strategies in which the nodal motions are independent, explicitly defined, and restricted to only nodes that lie on the moving boundary. The resulting mesh discrepancy volumes $|\mathcal{R}_h^n|$ are of order h , and the coefficients $C_1(u, v_h, T)$ and $C_2(u, v_h, T)$ can be shown to be of order $\ell_{h,r} h^{-1/2}$ and of order unity, respectively, for suitable nodal motions. The ultimate error estimate reads

$$\|u_h^{\Delta t, N} - u^N\|_{0,2,\Omega^T} \leq C(u, T)(\ell_{h,r} h^{r-1/2} + \Delta t^q + \ell_{h,r} h^{r+1/2} \Delta t^{-1}) \quad (23)$$

for a constant $C(u, T)$, which is suboptimal by half an order when $\Delta t \sim h$. Note that in practice, we have observed in numerical experiments that the use of a projector with stability constant $C_p > 1$ (such as the interpolation operator i_h^n) does not lead to a degradation of convergence beyond the existing half-order suboptimality, despite the theory's predictions.

5. Universal Meshes

The algorithm presented in the preceding section requires at each temporal node t^{n-1} the selection of a family of maps $\Phi_h^{n,t} : \mathcal{D}(\mathcal{S}_h^n) \rightarrow \Omega^t$, $t \in (t^{n-1}, t^n]$, from a fixed polygonal domain $\mathcal{D}(\mathcal{S}_h^n)$ to the moving domain Ω^t . Here we present a means of constructing such maps using a single, *universal mesh* that triangulates an ambient domain $\mathcal{D} \subset \mathbb{R}^2$ containing the domains $\{\Omega^t\}_{t=0}^T$ for all times $t \in [0, T]$. Full details of the method are described in [5].

The essence of the method is to triangulate \mathcal{D} with a fixed mesh \mathcal{T}_h and to identify, for each time interval $(t^{n-1}, t^n]$, a submesh \mathcal{S}_h^n of \mathcal{T}_h that approximates $\Omega^{t^{n-1}}$. Triangles on the boundary of \mathcal{S}_h^n are then deformed in such a way that the submesh conforms exactly to the moving domain Ω^t for all $t \in (t^{n-1}, t^n]$.

The conditions under which a given triangulation \mathcal{T}_h can be so adapted to conform to a family of domains Ω^t , $t \in [0, T]$, are laid forth in [61, 5]. Briefly, the procedure is guaranteed to succeed if:

- (i) Ω^t is C^2 -regular for every t .

- (ii) \mathcal{T}_h is sufficiently refined in a neighborhood of $\partial\Omega^t$ for every t .
- (iii) All triangles in \mathcal{T}_h have angles bounded above by a constant $\vartheta < \pi/2$.

The level of refinement requested by condition (ii) is dictated primarily by the minimum radius of curvature of $\partial\Omega^t$ among all times $t \in [0, T]$, which, roughly speaking, must be no less than a small multiple of the maximum element diameter. This notion is made precise in [61]. Note that condition (i) precludes an application of the method in its present form to domains with corners.

5.1. Construction of an Exactly Conforming Mesh

In detail, consider a triangulation \mathcal{T}_h of \mathcal{D} satisfying conditions (i-iii), with the parameter h denoting the length of the longest edge in the triangulation. For a given domain $\Omega^t \subset \mathcal{D}$, $t \in [0, T]$, let $\phi^t : \mathcal{D} \rightarrow \mathbb{R}$ denote the signed distance function to $\partial\Omega^t$, taken to be positive outside Ω^t and negative inside Ω^t . Let $\pi^t : \mathcal{D} \rightarrow \partial\Omega^t$ denote the closest point projection onto $\partial\Omega^t$. For $i = 0, 1, 2, 3$, let $\mathcal{T}_{h,i}^t$ denote the collection of triangles $K \in \mathcal{T}_h$ for which exactly i vertices of K do not lie in the interior of Ω^t .

For a given subtriangulation \mathcal{S}_h of \mathcal{T}_h , we make the distinction between \mathcal{S}_h , the list of vertices in the subtriangulation and their connectivities, and $\mathcal{D}(\mathcal{S}_h)$, the polygonal domain occupied by triangles in \mathcal{S}_h . We write $K \in \mathcal{S}_h$ to refer to triangles $K \subseteq \mathcal{D}(\mathcal{S}_h)$ who have vertices in \mathcal{S}_h .

To construct a conforming mesh for Ω^t from the mesh \mathcal{T}_h , we choose

$$\mathcal{S}_h^n = \mathcal{T}_{h,0}^{t^{n-1}} \cup \mathcal{T}_{h,1}^{t^{n-1}} \cup \mathcal{T}_{h,2}^{t^{n-1}}$$

as the reference subtriangulation for the domains Ω^t , $t \in (t^{n-1}, t^n]$. This subtriangulation is simply the set of triangles in \mathcal{T}_h with at least one vertex in $\Omega^{t^{n-1}}$. The map $\Phi_h^{n,t} : \mathcal{D}(\mathcal{S}_h^n) \rightarrow \Omega^t$ will then make use of three important mappings, described in the following paragraphs, and illustrated in Fig. 8. The *universal mesh map*, as described in [5], is $\Phi_h^{n,t^{n-1}}$.

Boundary evolution map. The first is a *boundary evolution map* $\gamma_h^{n,t} : \partial\mathcal{D}(\mathcal{S}_h^n) \rightarrow \partial\Omega^t$, which provides a correspondence between the piecewise linear boundary of $\mathcal{D}(\mathcal{S}_h^n)$ and the boundary of Ω^t for $t \in (t^{n-1}, t^n]$, as in Fig. 8. The choice of $\gamma_h^{n,t}$ is not unique, although a simple choice is the closest point projection onto Ω^t composed with the closest point projection onto $\Omega^{t^{n-1}}$:

$$\gamma_h^{n,t} = \pi^t \circ \pi^{t^{n-1}} \big|_{\partial\mathcal{D}(\mathcal{S}_h^n)}. \quad (24)$$

By the regularity of the spacetime domain Ω , this map is well-defined for h sufficiently small and t sufficiently close to t^{n-1} ; see [61].

Relaxation map. The second is a *relaxation map* $\mathbf{p}_h^{n,t}$ that perturbs vertices lying both inside Ω^t and near $\partial\Omega^t$ in a direction away from $\partial\Omega^t$. A simple choice of relaxation is the map

$$\mathbf{p}_h^{n,t}(x) = \begin{cases} x - \delta h \left(1 + \frac{\phi^{t^{n-1}}(x)}{Rh} \right) \nabla \phi^{t^{n-1}}(x) & \text{if } -Rh < \phi^{t^{n-1}}(x) < 0 \\ x & \text{otherwise,} \end{cases} \quad (25)$$

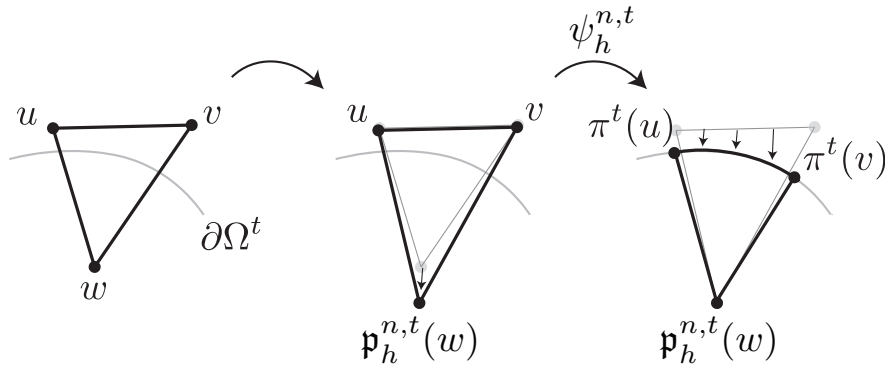


Figure 8: The action of $\Phi_h^{n,t}$ on a triangle $K \in \mathcal{T}_{h,2}^{t^{n-1}}$ comprises two steps: A relaxation step that moves w away from the boundary, and a nonlinear blend map $\psi_h^{n,t}$ that maps the straight triangle to a curved one.

which moves vertices within a distance Rh of $\partial\Omega^{t^{n-1}}$ by an amount $\leq \delta h$ in a direction normal to the boundary, with $R > 1$ a small positive integer and $(1 + 1/R)^{-1} \leq \delta \leq 1$. It is proven in [5] that for a straight boundary (or one of small enough radius of curvature compared with the mesh size) such a map results in elements of bounded quality at $t = t^{n-1}$ when conditions (i-iii) hold.

Note that this choice of relaxation leaves relaxed vertices fixed over the duration of the interval $(t^{n-1}, t^n]$. We denote by $\mathbf{p}_h^{n,t}(\mathcal{T}_h)$ the triangulation obtained by applying the relaxation $\mathbf{p}_h^{n,t}$ to the vertices of \mathcal{T}_h while preserving the mesh's connectivity.

Blend map. Finally, we will make use of a *blend map* $\psi_h^{n,t}$ which takes a straight triangle $K \in \mathbf{p}_h^{n,t}(\mathcal{T}_{h,2}^{t^{n-1}})$ to a curved triangle that conforms exactly to the boundary. The map we employ is proposed in [5]. Letting u, v, w denote the vertices of K , the blend map reads

$$\begin{aligned} \psi_h^{n,t}(x) = & \frac{1}{2(1 - \lambda_u)} [\lambda_v \gamma_h^{n,t}(\lambda_u u + (1 - \lambda_u)v) + \lambda_u \lambda_w \gamma_h^{n,t}(u)] \\ & + \frac{1}{2(1 - \lambda_v)} [\lambda_u \gamma_h^{n,t}((1 - \lambda_v)u + \lambda_v v) + \lambda_v \lambda_w \gamma_h^{n,t}(v)] + \lambda_w w, \end{aligned} \quad (26)$$

where $\lambda_u, \lambda_v, \lambda_w$ are the barycentric coordinates of $x \in K$. Here, we have employed the convention the vertex w is the unique vertex of K lying inside $\Omega^{t^{n-1}}$. It is not difficult to check that for fixed t , the blend map $\psi_h^{n,t}$ maps points x lying on the edge uv to their images under the boundary evolution map $\gamma_h^{n,t}$, preserves the location of the vertex w , and is affine on the edges wu and wv .

Culmination. We now define $\Phi_h^{n,t}$ over each triangle $K \in \mathcal{S}_h^n$ with vertices u, v, w according to

$$\Phi_h^{n,t}(x) = \begin{cases} \lambda_u \mathbf{p}_h^{n,t}(u) + \lambda_v \mathbf{p}_h^{n,t}(v) + \lambda_w \mathbf{p}_h^{n,t}(w) & \text{if } K \in \mathcal{T}_{h,0}^{t^{n-1}} \\ \lambda_u \gamma_h^{n,t}(u) + \lambda_v \mathbf{p}_h^{n,t}(v) + \lambda_w \mathbf{p}_h^{n,t}(w) & \text{if } K \in \mathcal{T}_{h,1}^{t^{n-1}} \\ \psi_h^{n,t}(\lambda_u u + \lambda_v v + \lambda_w \mathbf{p}_h^{n,t}(w)) & \text{if } K \in \mathcal{T}_{h,2}^{t^{n-1}}, \end{cases} \quad (27)$$

where $\lambda_u, \lambda_v, \lambda_w$ are the barycentric coordinates of $x \in K$. Once again, we have employed the convention that for triangles $K \in \mathcal{T}_{h,2}^{t^{n-1}}$, the vertex w is the unique vertex of K lying inside $\Omega^{t^{n-1}}$, and for triangles $K \in \mathcal{T}_{h,1}^{t^{n-1}}$, the vertex u is the unique vertex of K lying outside $\Omega^{t^{n-1}}$.

The domain evolution and its velocity. It is now straightforward to record explicit expressions for the domain mapping $\varphi_h^{n,t}$ and its material velocity $V_h^{n,t}$. By definition,

$$\varphi_h^{n,t} = \Phi_h^{n,t} \circ \left(\Phi_h^{n,t^{n-1}} \right)^{-1}. \quad (28)$$

The velocity field $V_h^{n,t}$ is then given by differentiation with respect to time:

$$V_h^{n,t} = \dot{\Phi}_h^{n,t} \circ \left(\Phi_h^{n,t^{n-1}} \right)^{-1}.$$

If the relaxation map $\mathbf{p}_h^{n,t}$ is independent of time over $(t^{n-1}, t^n]$ (as is the case for the choice (25)), this expression for $V_h^{n,t}$ is given explicitly by

$$V_h^{n,t}(X) = \begin{cases} 0 & \text{if } K \in \mathcal{T}_{h,0}^{t^{n-1}} \\ \lambda_u \dot{\gamma}_h^{n,t}(u) & \text{if } K \in \mathcal{T}_{h,1}^{t^{n-1}} \\ \frac{\lambda_v}{2(1-\lambda_u)} \dot{\gamma}_h^{n,t}(\lambda_u u + (1-\lambda_u)v) + \frac{\lambda_u \lambda_w}{2(1-\lambda_u)} \dot{\gamma}_h^{n,t}(u) & \text{if } K \in \mathcal{T}_{h,2}^{t^{n-1}}, \\ \quad + \frac{\lambda_u}{2(1-\lambda_v)} \dot{\gamma}_h^{n,t}((1-\lambda_v)u + \lambda_v v) + \frac{\lambda_v \lambda_w}{2(1-\lambda_v)} \dot{\gamma}_h^{n,t}(v) & \end{cases} \quad (29)$$

where $\lambda_u, \lambda_v, \lambda_w$ are the barycentric coordinates of $(\Phi_h^{n,t^{n-1}})^{-1}(X) \in K$, with the conventional ordering of the vertices described earlier. Formulas for the time derivative of π^t (which are needed for the choice $\gamma_h^{n,t} = \pi^t \circ \pi^{t^{n-1}}$) in terms of local measures of the boundary's shape and velocity are given in Appendix B.

5.2. Alternative: Isoparametric Approximation of the Domain

A convenient alternative to exact representations of the domain is to adopt superparametric or isoparametric representations of the domain. This entails approximating the map $\Phi_h^{n,t}$ (and hence the domain Ω^t) with a polynomial interpolant

$$\Phi_{h,\text{approx}}^{n,t}(\tilde{X}) = \sum_a \tilde{M}_a(\tilde{X}) \Phi_h^{n,t}(\tilde{Y}_a) \quad (30)$$

constructed from shape functions \tilde{M}_a of a triangular Lagrange element (henceforth termed Lagrange shape functions) with corresponding degrees of freedom \tilde{Y}_a on the reference triangulation \mathcal{S}_h^n . In this way, expressions for the spatial derivatives of the corresponding shape functions

$$N_{a,\text{approx}} = \tilde{N}_a \circ \left(\Phi_{h,\text{approx}}^{n,t^{n-1}} \right)^{-1} \quad (31)$$

and

$$n_{a,\text{approx}}^t = \tilde{N}_a \circ \left(\Phi_{h,\text{approx}}^{n,t} \right)^{-1}$$

involve only derivatives of the reference triangulation's shape functions \tilde{N}_a and the Lagrange shape functions \tilde{M}_a , and not the gradients of the exact map $\Phi_h^{n,t}$:

$$\begin{aligned}\nabla_X N_{a,\text{approx}}(X) &= \nabla_{\tilde{X}} \tilde{N}_a(\tilde{X}) \cdot \left(\nabla_{\tilde{X}} \Phi_{h,\text{approx}}^{n,t_+^{n-1}} \right)^{-1} = \nabla_{\tilde{X}} \tilde{N}_a(\tilde{X}) \cdot \left(\sum_a \nabla_{\tilde{X}} \tilde{M}_a(\tilde{X}) \Phi_h^{n,t_+^{n-1}}(\tilde{Y}_a) \right)^{-1} \\ \nabla_x n_{a,\text{approx}}^t(x) &= \nabla_{\tilde{X}} \tilde{N}_a(\tilde{X}) \cdot \left(\nabla_{\tilde{X}} \Phi_{h,\text{approx}}^{n,t} \right)^{-1} = \nabla_{\tilde{X}} \tilde{N}_a(\tilde{X}) \cdot \left(\sum_a \nabla_{\tilde{X}} \tilde{M}_a(\tilde{X}) \Phi_h^{n,t}(\tilde{Y}_a) \right)^{-1}.\end{aligned}$$

This, in turn, eliminates the need to compute gradients of the closest point projection π^t . This, and other reasons detailed later, make approximating the domain in this way more computationally convenient in practice.

For completeness, we next detail the corresponding approximate domain map

$$\varphi_{\text{approx}}^{n,t} = \Phi_{h,\text{approx}}^{n,t} \circ \left(\Phi_{h,\text{approx}}^{n,t_+^{n-1}} \right)^{-1}$$

and velocity fields, which take particularly simple forms. In fact, with

$$y_a(t) = \Phi_h^{n,t}(\tilde{Y}_a)$$

denoting the trajectory of a degree of freedom \tilde{Y}_a and

$$M_a = \tilde{M}_a \circ \left(\Phi_{h,\text{approx}}^{n,t_+^{n-1}} \right)^{-1}$$

denoting the pushforward of the Lagrange shape functions \tilde{M}_a to $\Omega^{t^{n-1}}$, we have

$$\begin{aligned}\varphi_{h,\text{approx}}^{n,t}(X) &= \Phi_{h,\text{approx}}^{n,t} \left(\left(\Phi_{h,\text{approx}}^{n,t_+^{n-1}} \right)^{-1}(X) \right) \\ &= \sum_a \tilde{M}_a \left(\left(\Phi_{h,\text{approx}}^{n,t_+^{n-1}} \right)^{-1}(X) \right) \Phi_h^{n,t}(\tilde{Y}_a) \\ &= \sum_a M_a(X) y_a(t).\end{aligned}$$

The corresponding material and spatial velocity fields are thus

$$V_{h,\text{approx}}^{n,t}(X) = \sum_a M_a(X) \dot{y}_a(t)$$

and

$$v_{h,\text{approx}}^{n,t}(x) = \sum_a m_a^t(x) \dot{y}_a(t),$$

respectively, with $m_a^t = \tilde{M}_a \circ \left(\Phi_{h,\text{approx}}^{n,t} \right)^{-1}$.

Introducing approximations of the domain requires some extra care in the imposition of boundary conditions. In the example problem here, homogeneous Dirichlet boundary

conditions are imposed on the boundary of the approximate domain. Therefore, the order of the Lagrange shape functions \tilde{M}_a should be high enough to ensure that the errors introduced by approximating Ω^t with $\Phi_{h,\text{approx}}^{n,t}(\mathcal{S}_h^n)$ converge to zero at least as quickly as the error in the original spatial discretization as $h \rightarrow 0$. It is well-known [62] that if the shape functions \tilde{N}_a are themselves Lagrange shape functions, then it suffices to use Lagrange shape functions \tilde{M}_a of equal or higher degree for the approximation of the domain geometry. Elements of this type are referred to as *isoparametric* or *superparametric* elements, depending upon whether the functions \tilde{M}_a have equal or higher degree, respectively, than the functions \tilde{N}_a .

5.3. Example: A Complete Algorithm

We now present an algorithm that takes advantage of the SDIRK method of §4.3 for time integration and of the isoparametric representation of the domain of §5.2. For concreteness, we consider the case in which the partial differential operator $a(u) = -\Delta_x u$, so that

$$a^t(u, w) = \int_{\Omega^t} \nabla_x u \cdot \nabla_x w \, dx,$$

In what follows, we denote matrices and vectors with uppercase and lowercase boldface letters, respectively. As shorthand notation, we denote by

$$(u, w)_{K^t} = \int_{K^t} u(x)w(x) \, dx$$

the inner product of two functions u and w over an element $K^t = \Phi_h^{n,t}(K)$, $K \in \mathcal{S}_h^n$. The algorithm is labeled Algorithm 5.1.

Implementation. We discuss some key steps of the algorithm next, to show how the motion of the domain is accounted for in the implementation of the algorithm, and how it affects the computation of elemental quantities such as the mass matrix. For concreteness, in the following it is useful to keep in mind a very simple example, such as when the moving domain Ω^t is the circle centered at the origin of radius $1+t$, for each small $t \geq 0$ (this is the geometry used to draw Fig. 9 later). Without loss of generality, we discuss the case in which $n = 1$, so that $t^{n-1} = 0$. Finally, we will also use the standard triangle \hat{K} , such as that with vertices $(0, 0)$, $(0, 1)$, and $(1, 0)$, which has traditionally been used in finite element codes to perform quadrature.

In step 2 we identify triangles in $\mathcal{T}_{h,i}^0$, for $i = 0, 1, 2$, by labeling vertices of triangles in the universal mesh according to whether they are inside or outside Ω^0 . For example, Fig. 9 shows one triangle $\tilde{K} \in \mathcal{T}_{h,2}^0$. For this example \tilde{K} will be assumed to be quadratic and hence consists of 6 nodes, with its nodes labeled by \tilde{Y}_a , $a = 1, \dots, 6$.

In step 3, the positions $\{Y_a\}_a$ of these six nodes in the mesh conforming to Ω_0 are computed, and in general, of all nodes in triangles intersecting $\partial\Omega_0$. This computation involves computing the closest point projection for nodes $\tilde{Y}_1, \tilde{Y}_2, \tilde{Y}_3$, and \tilde{Y}_5 , moving the first 3 nodes to their closest point projections, and moving \tilde{Y}_5 along the normal to the boundary emanating from its closest point projection, according to (25). Nodes \tilde{Y}_4 and \tilde{Y}_6 are then mapped to the midpoints of segments Y_3Y_5 and Y_1Y_5 , respectively. These six nodes define the isoparametric quadratic triangle $K = \Phi_{h,\text{approx}}^{1,0+}(\tilde{K})$. Henceforth, *the construction of shape*

Algorithm 5.1 Time integration using a universal mesh with an s -stage SDIRK method

Require: Initial condition $u^0 \in H_0^1(\Omega^0)$.

1: **for** $n = 1, 2, \dots, N$ **do**

2: Identify triangles in $\mathcal{T}_{h,i}^{t^{n-1}}$, $i = 0, 1, 2$. Set $\mathcal{S}_h^n = \mathcal{T}_{h,0}^{t^{n-1}} \cup \mathcal{T}_{h,1}^{t^{n-1}} \cup \mathcal{T}_{h,2}^{t^{n-1}}$.

3: Compute $\Phi_h^{n,t_+^{n-1}}(\tilde{Y}_a)$ for every degree of freedom $\tilde{Y}_a \in \mathcal{D}(\mathcal{S}_h^n)$ using (27).

4: Set $\mathcal{V}_h^n = \text{span}\{N_{a,\text{approx}}\}_a$, where $\{N_{a,\text{approx}}\}_a$ are the shape functions (31).

5: Project $u_h^{\Delta t, n-1} \in (\varphi_h^{n-1, t^{n-1}})_* \mathcal{V}_h^{n-1}$ (or u^0 if $n = 1$) onto \mathcal{V}_h^n using a projector p_h^n . Denote by \mathbf{u}_0 the vector of coefficients in the expansion

$$p_h^n u_h^{\Delta t, n-1} = \sum_a (\mathbf{u}_0)_a N_{a,\text{approx}}.$$

6: **for** $i = 1, 2, \dots, s$ **do**

7: Compute $\Phi_h^{n,t_i}(\tilde{Y}_a)$ for every degree of freedom $\tilde{Y}_a \in \mathcal{D}(\mathcal{S}_h^n)$ using (27).

8: With $K^{t_i} = \Phi_{h,\text{approx}}^{t_i}(K)$ for every $K \in \mathcal{S}_h^n$, assemble

$$\begin{aligned} \mathbf{M}_{ab} &= \sum_K (n_{b,\text{approx}}^{t_i}, n_{a,\text{approx}}^{t_i})_{K^{t_i}} \\ \mathbf{B}_{ab} &= \sum_K (v_{h,\text{approx}}^{n,t_i} \cdot \nabla_x n_{b,\text{approx}}^{t_i}, n_{a,\text{approx}}^{t_i})_{K^{t_i}} \\ \mathbf{K}_{ab} &= \sum_K (\nabla_x n_{b,\text{approx}}^{t_i}, \nabla_x n_{a,\text{approx}}^{t_i})_{K^{t_i}} \\ \mathbf{f}_a &= \sum_K (f(t_i), n_{a,\text{approx}}^{t_i})_{K^{t_i}} \end{aligned}$$

9: With $\mathbf{u}_* = \sum_{j=0}^{i-1} \beta_{ij} \mathbf{u}_j$ and $\Delta t^n = t^n - t^{n-1}$, define

$$\begin{aligned} \mathbf{A} &= \mathbf{M} + \gamma \Delta t^n (\mathbf{K} - \mathbf{B}) \\ \mathbf{b} &= \mathbf{M} \mathbf{u}_* + \mathbf{f} \end{aligned}$$

10: For every degree of freedom $\tilde{Y}_a \notin \text{int}(\mathcal{D}(\mathcal{S}_h^n))$, set

$$\begin{aligned} \mathbf{A}_{ab} &= \delta_{ab} \\ \mathbf{b}_a &= 0. \end{aligned}$$

11: Solve $\mathbf{A} \mathbf{u}_i = \mathbf{b}$ for \mathbf{u}_i .

12: **end for**

13: Set $u_h^{\Delta t, n}(x) = \sum_a (\mathbf{u}_s)_a n_{a,\text{approx}}^{t^n}(x)$.

14: **end for**

15: **return** $u_h^{\Delta t, N}$

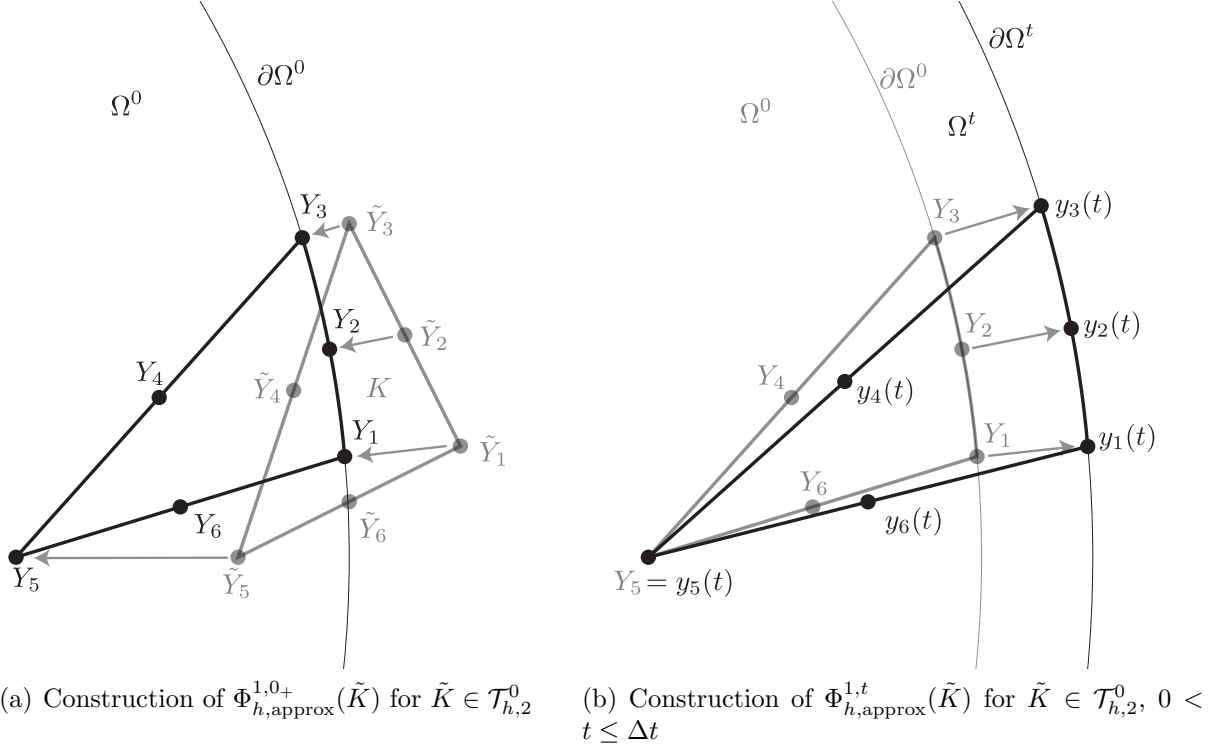


Figure 9: Example of how the approximate evolving domain is accounted for in practice. See text in §5.3 for the explanation.

functions and quadrature rules follow standard finite element procedures over isoparametric elements. For example, in this case, $K = \hat{\Psi}(\hat{K})$, where the isoparametric map is $\hat{\Psi}(\hat{X}) = \sum_{a=1}^6 Y_a \hat{N}_a(\hat{X})$ and $\{\hat{N}_a\}_a$ denote the shape functions over \hat{K} . This is equivalent to (30).

In step 4, the shape functions over K are constructed. Because the map between \hat{K} and \tilde{K} is affine, the shape function $\{N_{a,approx}\}_a$ can be constructed over \hat{K} , namely, $N_{a,approx}(X) = \hat{N}_a(\hat{\Psi}^{-1}(X))$, for $X \in K$. This is, again, standard procedure for isoparametric elements.

As the boundary of the domain moves at each stage i of the time integration, the nodes $\{Y_a\}_a$ of triangle K are deformed as follows (step 7): Nodes Y_1, Y_2, Y_3 are mapped to their closest point projections onto $\partial\Omega^{t_i}$, labeled y_1, y_2 , and y_3 , respectively, node Y_5 remains where it is, so $y_5 = Y_5$, and nodes Y_4 and Y_6 are mapped to y_4 and y_6 , the midpoints of edges y_3y_5 and y_1y_5 , respectively, see Fig. 9b. Shape functions over triangle K^{t_i} are formed in precisely the same way as those for triangle K , in this case with nodal positions $\{y_a\}_a$.

To assemble the system needed to solve (20) at each stage of the time integration (step

8) it is useful to notice that the elemental mass matrix for each element is computed as

$$\begin{aligned}
\mathbf{M}_{ab}^K &= \int_{K^{t_i}} n_{a,\text{approx}}^{t_i} n_{b,\text{approx}}^{t_i} dx \\
&= \int_K N_{a,\text{approx}} N_{b,\text{approx}} |\nabla_X \varphi_{h,\text{approx}}^{1,t_i}| dX \\
&= \int_{\tilde{K}} \tilde{N}_a \tilde{N}_b |\nabla_{\tilde{X}} \Phi_{h,\text{approx}}^{1,t_i}| d\tilde{X} \\
&= \int_{\hat{K}} \hat{N}_a \hat{N}_b |\nabla_{\hat{X}} \hat{\Psi}| d\hat{X},
\end{aligned} \tag{32}$$

and similarly for the elemental contributions to the other terms of (20). Consequently, quadrature could be performed on any of the triangles K , \tilde{K} , \hat{K} , or K^{t_i} , but for convenience and following standard practice, we do it over the standard element \hat{K} .

Notice then that, in order to perform the quadrature over \hat{K} , it is convenient to build the deformed mesh at time t_i , since it makes the construction of $\hat{\Psi}$ straightforward. So, in this time-integration scheme the deformed mesh is built s times in a time step.

An important practical matter we wish to highlight is the simplicity of the data structures needed to implement our method. In particular, *the connectivity of the universal mesh never changes* during deformation – only the nodal positions change. As a consequence, the sizes and sparsity structures of various discrete quantities (the solution vector \mathbf{u} , the mass matrix \mathbf{M} , the stiffness matrix \mathbf{K} , the convection matrix \mathbf{B} , and the forcing vector \mathbf{f}) can be held fixed, even though differing subsets of degrees of freedom may participate in the discrete equations at different intervals $(t^{n-1}, t^n]$. This can be accomplished by simply imposing “homogeneous Dirichlet boundary conditions” on the solution at degrees of freedom not belonging to the subtriangulation \mathcal{S}_h^n . In practice, this amounts to replacing the corresponding rows of a particular matrix \mathbf{A} with rows whose only nonzero entries are 1 on the diagonal, and setting to zero the corresponding entries of a vector (see step 10 of Algorithm 5.1). Note that \mathbf{A} is automatically asymmetric at the outset, so any concerns of breaking symmetry via row replacement are irrelevant.

5.4. Exact vs. Approximate Map: Cost Considerations

The computational cost of evaluating the map $\Phi_h^{n,t}$ or its approximant $\Phi_{h,\text{approx}}^{n,t}$ is dominated by the cost of evaluating closest-point projections onto $\partial\Omega^t$. In our numerical experiments (which used $\Phi_{h,\text{approx}}^{n,t}$), these calculations accounted for little more than 5 – 10% of the total run time of a typical simulation.

Note that implementations that employ the exact map $\Phi_h^{n,t}$ require evaluations of the closest point projection and its gradient at *quadrature points* in triangles $K \in \mathcal{T}_{h,1}^{t^{n-1}} \cup \mathcal{T}_{h,2}^{t^{n-1}}$, whereas implementations that employ the approximate map $\Phi_{h,\text{approx}}^{n,t}$ require evaluations only of the closest point projection (not its gradient) on those triangles’ *degrees of freedom*. A counting argument reveals that the computational savings that accompany the use of $\Phi_{h,\text{approx}}^{n,t}$ over $\Phi_h^{n,t}$ are significant: For a polynomial interpolant $\Phi_{h,\text{approx}}^{n,t}$ constructed from Lagrange elements of a fixed polynomial degree, it is not difficult to show that the use of $\Phi_{h,\text{approx}}^{n,t}$ over $\Phi_h^{n,t}$ reduces the computational cost (measured by number of closest point projection evaluations) by factors of 9, 9, and 5.2, respectively, for affine, quadratic, and cubic

Lagrange elements, assuming the use of a quadrature rule that exactly computes entries of the elemental mass matrix on straight triangles.

5.5. Error estimate for a universal mesh

We conclude this section by applying the error estimate (21) to the case in which the maps $\Phi_h^{n,t}$ are constructed from a universal mesh according to the algorithm in Section 5.1.

It is shown in our forthcoming paper [6] that for u and Ω sufficiently regular, there exist constants B , c , C , $\bar{C}_1(u, T)$, and $\bar{C}_2(u, T)$, independent of h , N , and Δt , such that

$$C_1(u, v_h, T) \leq \ell_{h,r} h^{-1/2} \bar{C}_1(u, T) \quad (33)$$

$$C_2(u, v_h, T) \leq \bar{C}_2(u, T) \quad (34)$$

$$N \leq B T \Delta t^{-1} \quad (35)$$

$$|\mathcal{R}_h^n| \leq C h, \quad n = 1, 2, \dots, N \quad (36)$$

as long as

$$c \max_{1 \leq n \leq N} (t^n - t^{n-1}) \leq h \quad (37)$$

and

$$\max_{1 \leq n \leq N} (t^n - t^{n-1}) \leq B \min_{1 \leq n \leq N} (t^n - t^{n-1}),$$

where $\ell_{h,r}$ is given by (22). We remind the reader that $\Delta t \leq \min_{1 \leq n \leq N} (t^n - t^{n-1})$.

Relations (33-36) imply that the numerical solution obtained from the proposed strategy has the accuracy stated in (23), that is,

$$\|u_h^{\Delta t, N} - u^N\|_{0,2,\Omega^T} \leq C(u, T) (\ell_{h,r} h^{r-1/2} + \Delta t^q + \ell_{h,r} h^{r+1/2} \Delta t^{-1}). \quad (38)$$

When Δt and h scale proportionately, this convergence rate is suboptimal by half an order (up to a logarithmic factor if $r = 2$).

Time step restriction. Notice that the text above included the restriction (37), which implies a restriction on the time step of the form $c \Delta t \leq h$. The necessity of such a restriction is made clear by noting that the mesh motion defined by $\varphi_h^{n,t}$, $t \in (t^{n-1}, t^n]$, leaves all elements stationary except those with an edge on the moving boundary. Imposing (37) with a suitable choice of c ensures that the image under $\varphi_h^{n,t}$ of each such element has an aspect ratio that is bounded above and below uniformly in h for all times $t \in (t^{n-1}, t^n]$. In particular, it ensures that no element collapses to a set of nonpositive measure at any time $t \in (t^{n-1}, t^n]$. Note that this restriction is intrinsic to the mesh motion strategy; it is a restriction that must be imposed in addition to any time step restriction needed to ensure stability of the particular time integrator chosen.

Explanation of estimate. Let us briefly describe how the dependencies (33-36) arise. To begin, consider the regularity of the velocity field v_h . The proposed strategy employs a velocity field that is of order unity on the boundary of Ω^t and decays to zero over a strip of neighboring elements, i.e. a strip of width $O(h)$. It follows that the velocity field itself is everywhere of order unity, but its spatial gradient is of order h^{-1} . Moreover, the support of $v_h^{n,t}$ and its derivatives (the strip of neighboring elements) has measure $O(h)$. From these

observations, a simple calculation reveals that the L^2 norm of $\nabla_x v_h^{n,t}$ is of order $h^{-1/2}$. It is this fact that contributes to the dependence of $C_1(u, v_h, T)$ on $h^{-1/2}$. The factor $\ell_{h,r}$ arises because of the approximation properties in the L^∞ -norm of piecewise linear finite element spaces.

Consider next the relation (36). The quantity $|\mathcal{R}_h^n|$ measures the discrepancy between two triangulations of the same domain $\Omega^{t^{n-1}}$, namely $\Phi_h^{n-1,t^{n-1}}(\mathcal{S}_h^{n-1})$ and $\Phi_h^{n,t^{n-1}}(\mathcal{S}_h^n)$. By (25) and (27), these triangulations differ only in a neighborhood of $\partial\Omega^{t^{n-1}}$ that has measure $O(h)$. A consequence of this fact is that the contribution to the error associated with projecting the solution onto a new finite element space at each temporal node t^n is of order $Nh^{r+1/2} \sim h^{r+1/2}\Delta t^{-1}$, rather than of the order $Nh^r \sim h^r\Delta t^{-1}$ that one might expect if the triangulations had differed over the entire domain.

Notice that the terms $\ell_{h,r}h^{r-1/2}$ and $\ell_{h,r}h^{r+1/2}\Delta t^{-1}$ in (38) are balanced when $\Delta t \sim h$. Roughly speaking, the asymptotically unbounded gradient of $v_h^{n,t}$ introduces a half-order reduction in the spatial discretization error (ordinarily h^r), but the small support of $v_h^{n,t}$ mitigates the reduction of order introduced by the repeated projections (ordinarily a full order) to half an order.

Optimal balance. Given the half-order reduction in error associated with the use of a velocity field $v_h^{n,t}$ whose support has measure $O(h)$, it is tempting to consider the possibility of employing mesh motions with more broadly supported velocity fields. An immediate consequence of such a decision, however, is an increase in the error $h^r\ell_{h,r}|\mathcal{R}_h^n|^{1/2}$ associated with changing finite element spaces. Indeed, if the triangulations $\Phi_h^{n-1,t^{n-1}}(\mathcal{S}_h^{n-1})$ and $\Phi_h^{n,t^{n-1}}(\mathcal{S}_h^n)$ differ over a region of measure $O(1)$, then $|\mathcal{R}_h^n|$ is of order 1 rather than of order h . The resulting total error estimate is suboptimal by *one* order rather than half an order when $N = O(\Delta t^{-1})$. For this reason, the strategy as it has been presented at the outset can be said to provide an optimal balance between competing sources of error.

Note, however, that if the finite element spaces are changed less frequently (i.e. $N = O(1)$), then optimal order accuracy is, in principle, obtainable via the use of more broadly supported velocity fields. This is precisely what is accomplished by conventional ALE schemes, which adopt global mesh motion strategies (in which all nodes of the mesh participate) and remesh occasionally (at temporal nodes t^n whose separation in time is of order unity). The price to be paid for such a decision, of course, is a reduction in the efficiency and robustness of the mesh motion strategy. Conventional ALE mesh motions commonly require the solution of systems of equations (such as those of linear elasticity) for the positions of mesh nodes [36, 37, 38, 39]; in contrast, the nodal motions in our method are independent and explicitly defined, rendering the mesh motion strategy low-cost and easily parallelizable. Second, our mesh motion strategy is robust in the sense that it enjoys provable bounds on the quality of the deformed mesh under suitable constraints on the time step and mesh spacing [5, 6].

6. Numerical Examples

In this section, we apply the proposed method to a modification of a classical moving-boundary problem: Stefan's problem. In our modification, the evolution of the boundary is imposed through the exact solution, instead of being computed. Our aim in this example is

is to illustrate the convergence rate of the method with respect to the mesh spacing h and time step Δt .

We begin by demonstrating, using a one-dimensional numerical test, that the bound (38) is sharp. That is, the order of accuracy of the method is suboptimal by half an order in the L^2 norm when Δt and h scale proportionately. We observe, however, that the suboptimal rate is difficult to detect from an inspection of the total error $\|u_h^{N,\Delta t} - u^N\|_{0,2,\Omega^T}$, since the terms of suboptimal order contributing to the total error are dominated by terms of optimal order (for practical values of the mesh spacing h). We follow with a convergence test in two-dimensions, where, for the reason just described, optimal rates are observed for the total error.

6.1. The (Modified) One-Dimensional Stefan Problem with Prescribed Boundary Evolution

Consider the following instance of the one-dimensional Stefan problem: Find $u(x, t)$ and $s(t)$ such that

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}, \quad 0 < x < s(t), \quad t \geq 1 \quad (39a)$$

$$\frac{ds}{dt} = -\frac{\partial u}{\partial x}, \quad x = s(t) \quad (39b)$$

$$u(0, t) = e^t - 1, \quad t \geq 1 \quad (39c)$$

$$u(s(t), t) = 0, \quad t \geq 1 \quad (39d)$$

$$u(x, 1) = e^{1-x} - 1, \quad 0 \leq x \leq 1 \quad (39e)$$

$$s(1) = 1. \quad (39f)$$

The exact solution is

$$\begin{aligned} u(x, t) &= e^{t-x} - 1 \\ s(t) &= t. \end{aligned}$$

In this case, we treat the boundary evolution as prescribed by supplying the exact evolution $s(t)$, instead of solving for it by integrating (39b).

We computed the numerical solution $u_h^{\Delta t, N}$ using a finite element space made of continuous elementwise-affine functions on a sequence of uniform meshes with spacing $h = 2^{-k}h_0$, $k = 0, 1, 2, 3$ over the time interval $[1, T]$ with $h_0 = 1/4$ and $T = 1 + 10^{-6}$ (the short time interval was chosen, on the basis of numerical experiments, in order to detect the suboptimal rate predicted by the theory). The restriction of the algorithm to a single spatial dimension is that specified in Algorithm 2.1, and is complemented with the choice $p_h^n = p_{h,L^2}^n$ for the projection, relaxation parameters $\delta = 0.3$ and $R = 3$, and the singly diagonally implicit Runge-Kutta (SDIRK) scheme of order 2 given in Table A.3 with a time step $\Delta t = 10^{-6}h/h_0$ for time-integration.

Table 1 presents the convergence of the method measured at time $t = T$ in $L^2(\Omega^T)$. The third column of the table suggests that the total error $\|u_h^{\Delta t, N} - u^N\|_{0,2,\Omega^T}$ converges at an optimal rate $O(h^2)$. However, columns 1 and 2 reveal that a piece of the error, namely the discrepancy between the numerical solution $u_h^{\Delta t, N}$ and the nodal interpolant $i_h^{t^N} u^N$ of

Table 1: Convergence rates in the L^2 -norm on Ω^T for the solution to the (modified) one-dimensional Stefan problem using a finite element space made of continuous elementwise-affine functions with a second-order implicit Runge-Kutta time integrator, see §6.1. Differences between the exact solution u^N , the numerical approximation $u_h^{\Delta t, N}$, and the nodal interpolant of the exact solution $i_h^{t^N} u^N$ are shown in each column. These values are used in §6.1 to illustrate that the expected theoretical convergence rate of $h^{3/2}$ is observed. Nevertheless, the slowly converging part is so small, that the apparent convergence rate is h^2 , as the third column shows.

h_0/h	$\ u_h^{\Delta t, N} - i_h^{t^N} u^N\ _{0,2,\Omega^T}$	Order	$\ i_h^{t^N} u^N - u^N\ _{0,2,\Omega^T}$	Order	$\ u_h^{\Delta t, N} - u^N\ _{0,2,\Omega^T}$	Order
1	1.2e-11	-	4.4e-05	-	4.4e-05	-
2	4.6e-12	1.42	1.1e-05	2.04	1.1e-05	2.04
4	1.6e-12	1.49	2.6e-06	2.02	2.6e-06	2.02
8	5.5e-13	1.56	6.5e-07	2.01	6.5e-07	2.01

Table 2: Convergence rates in the L^2 -norm on Ω^T for the solution to the (modified) two-dimensional Stefan problem (40) using linear, quadratic, and cubic elements together with nodal interpolation as the projection operator, and second-, third-, and fourth-order implicit Runge-Kutta schemes, respectively, as time integrators. See Fig. 10 for a graphical depiction of the same results.

h_0/h	Linear		Quadratic		Cubic	
	Error	Order	Error	Order	Error	Order
1	3.0e-02	-	1.3e-03	-	2.9e-05	-
2	9.8e-03	1.59	1.4e-04	3.21	3.1e-06	3.24
4	2.6e-03	1.94	2.1e-05	2.66	2.2e-07	3.84
8	6.4e-04	2.00	2.6e-06	3.03	1.4e-08	3.97
16	1.6e-04	2.00	3.3e-07	2.97	-	-

the exact solution, decays at a suboptimal rate $O(h^{3/2})$. Since standard estimates from the theory of interpolation give $\|u^N - i_h^{t^N} u^N\|_{0,2,\Omega^T} = O(h^2)$, it follows from the inequality

$$\|u_h^{\Delta t, N} - i_h^{t^N} u^N\|_{0,2,\Omega^T} \leq \|u_h^{\Delta t, N} - u^N\|_{0,2,\Omega^T} + \|u^N - i_h^{t^N} u^N\|_{0,2,\Omega^T}$$

that $\|u_h^{\Delta t, N} - u^N\|_{0,2,\Omega^T}$ must be decaying no faster than $O(h^{3/2})$. However, the contribution to the error supplied by $u_h^{\Delta t, N} - i_h^{t^N} u^N$ is several orders of magnitude smaller than the remaining contribution, $i_h^{t^N} u^N - u^N$, explaining the apparent optimal rate observed for the total error.

6.2. The (Modified) Two-Dimensional Stefan Problem with Prescribed Boundary Evolution

We consider now the following instance of the two-dimensional, cylindrically symmetric Stefan problem with a circular boundary of radius $\rho(t)$ centered at the origin. Find the

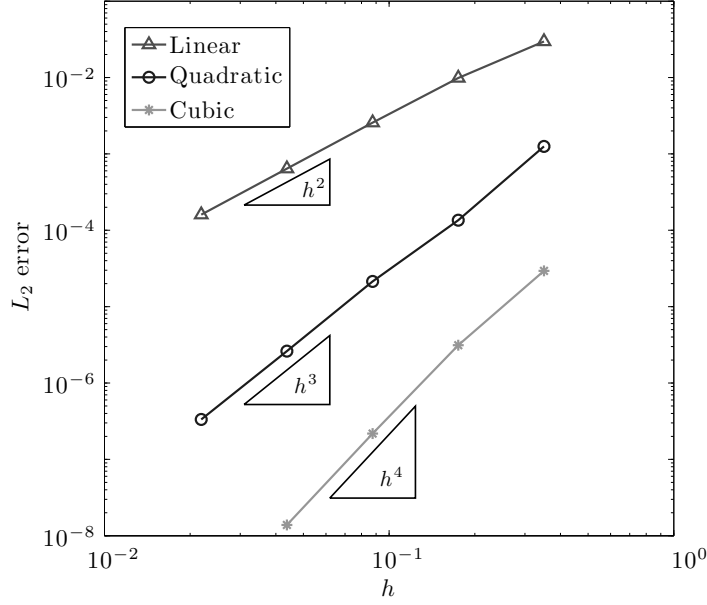


Figure 10: (a) L^2 -error $\|u_h^{\Delta t, N} - u^N\|_{0,2,\Omega^T}$ at a fixed final time as a function of the mesh spacing h for the (modified) two-dimensional Stefan problem (40) with prescribed boundary evolution. The problem was solved using linear, quadratic, and cubic elements together with nodal interpolation as the projection operator, and second-, third-, and fourth-order implicit Runge-Kutta schemes, respectively, as time integrators, with $h \propto \Delta t$.

scalar functions $u(x, t)$ and $\rho(t)$ such that for all times $t \in [0, T]$,

$$\frac{\partial u}{\partial t} - \Delta_x u = f, \quad 0 \leq |x| < \rho(t) \quad (40a)$$

$$\frac{d\rho}{dt} = -\frac{\partial u}{\partial n}, \quad |x| = \rho(t) \quad (40b)$$

$$u(x, t) = 0, \quad |x| = \rho(t) \quad (40c)$$

$$u(x, 0) = J_0(r_0|x|), \quad (40d)$$

$$\rho(0) = 1, \quad (40e)$$

where J_0 is the zeroth-order Bessel function of the first kind, r_0 is the smallest positive root of J_0 , and

$$\begin{aligned} f(x, t) &= \frac{\alpha r_0^3 \beta(t)^2 |x|}{2\sigma(t)^3} J'_0\left(\frac{r_0|x|}{\sigma(t)}\right) \\ \sigma(t) &= \exp\left(\frac{\alpha(\beta(t) - 1)}{2}\right) \\ \beta(t) &= \frac{1}{\alpha} \text{Ei}^{-1}(\text{Ei}(\alpha) - r_0^2 t e^\alpha) \\ \alpha &= \frac{2J'_0(r_0)}{r_0}. \end{aligned}$$

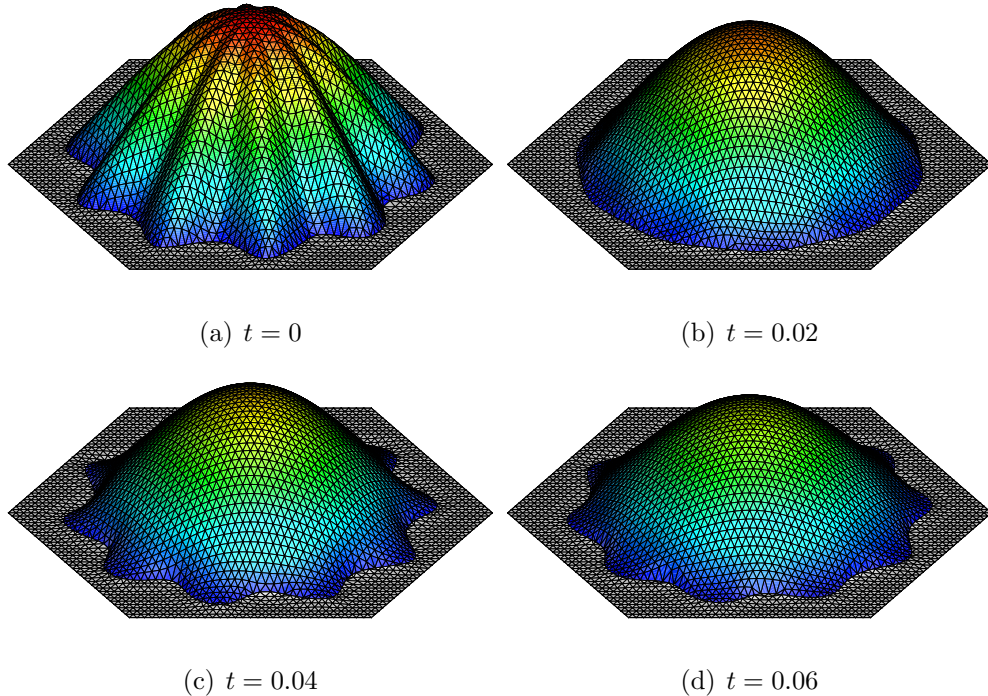


Figure 11: Solution to a prescribed-boundary variant of the Stefan problem in which the moving boundary is a sinusoidal perturbation of the unit circle.

Here, $\text{Ei}(z) = -\int_{-z}^{\infty} \frac{e^{-\zeta}}{\zeta} d\zeta$, the exponential integral. The exact solution is

$$u(x, t) = \beta(t) J_0 \left(\frac{r_0 |x|}{\sigma(t)} \right)$$

$$\rho(t) = \sigma(t).$$

In our implementation, we treat the boundary evolution as prescribed by supplying the exact evolution of the moving domain's radius $\rho(t)$, instead of solving for it. To study the convergence of the method, the problem was solved using finite element spaces of continuous functions that are affine, quadratic, and cubic over each element (linear, quadratic, and cubic Lagrange elements) together with nodal interpolation as the projection operator, relaxation parameters $\delta = 0.8$ and $R = 3$, and singly diagonally implicit Runge-Kutta (SDIRK) schemes of orders 2, 3, and 4, respectively, as the time integrators (see the coefficients in Tables A.3-A.5). The solution was computed on a uniform mesh of equilateral triangles with a lowest resolution mesh spacing of $h_0 = 0.35$ and a time step $\Delta t = Th/h_0$, up to a final time $T = 0.005$.

Fig. 10 displays the L^2 -error of the numerical solution as a function of the mesh spacing h at $t = T$. Optimal convergence orders of 2, 3, and 4 are observed for the three schemes, in agreement with the observations made in the one-dimensional test case. Table 2 shows the same results.

To illustrate the method on a second, more interesting example, we solved the partial differential equation (40a) with homogeneous Dirichlet boundary conditions and initial con-

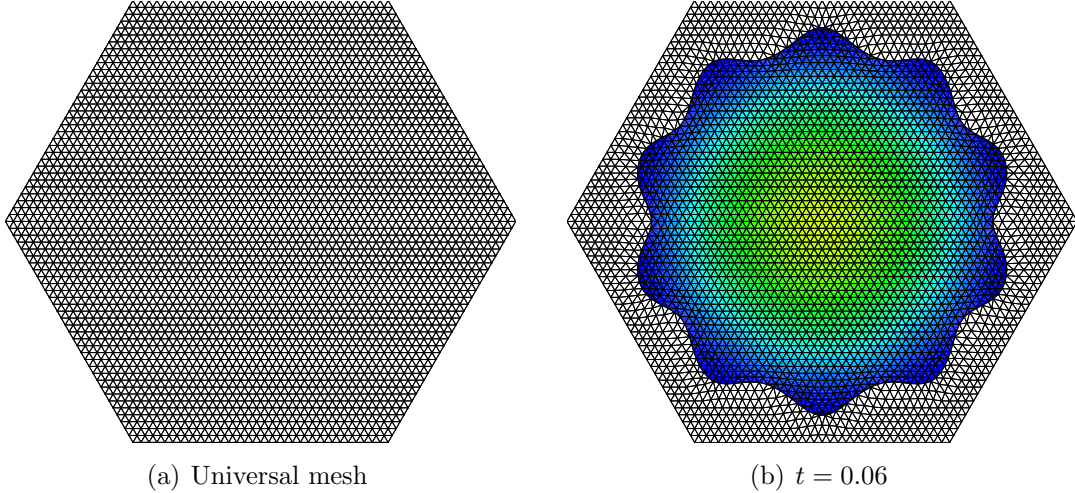


Figure 12: (a) Universal mesh adopted during the simulation depicted in Fig. 11, and (b) its image under the universal mesh map at $t = 0.06$, superposed with the contours of the solution.

dition

$$u(x) = J_0 \left(\frac{10r_0|x|}{10 + \cos 10\theta} \right)$$

on a prescribed domain Ω^t whose boundary is given by a sinusoidal perturbation of the unit circle. Namely,

$$\Omega^t = \left\{ x \mid |x| < 1 + \frac{1}{10} \cos 10\theta \cos 250t \right\}$$

with $\theta = \tan^{-1}(x_2/x_1)$. Fig. 11 shows snapshots of the solution, which was computed using quadratic Lagrange elements on a uniform mesh of equilateral triangles ($h = 0.04375$) together with nodal interpolation as the projection operator, relaxation parameters $\delta = 0.8$ and $R = 3$, and the third-order SDIRK scheme (A.4) with time step $\Delta t = 0.000625$. The universal mesh and its image under the universal mesh map at an instant in time are shown in Fig. 12.

7. Conclusion

We have presented a general framework for the design of high-order finite element methods for moving boundary problems with prescribed boundary evolution. A key role in our approach was played by universal meshes, which combine the immunity to large mesh distortions enjoyed by conventional fixed-mesh methods with the geometric fidelity of deforming-mesh methods. A given accuracy in space and time may be achieved by choosing an appropriate finite element space on the universal mesh and an appropriate time integrator for ordinary differential equations. The order of accuracy of the resulting scheme is suboptimal by one half an order according to theory, although we observed in our numerical examples that terms of optimal order tend to dominate in practice.

Several aspects of this research motivate further study. First, we have yet to address problems for which the boundary itself is an unknown, rather than prescribed. An extension

of the method to three dimensions is alluring. Finally, developing an analogous strategy for domains with lower regularity, such as domains with corners, is an open problem.

8. Acknowledgments

This research was supported by the U.S. Department of Energy grant DE-FG02-97ER25308; Department of the Army Research Grant, grant number: W911NF-07- 2-0027; and NSF Career Award, grant number: CMMI-0747089.

9. References

- [1] P. H. Saksono, W. G. Dettmer, D. Perić, An adaptive remeshing strategy for flows with moving boundaries and fluid–structure interaction, *International Journal for Numerical Methods in Engineering* 71 (9) (2007) 1009–1050.
- [2] F. Gibou, R. Fedkiw, A fourth order accurate discretization for the Laplace and heat equations on arbitrary domains, with applications to the Stefan problem, *Journal of Computational Physics* 202 (2005) 577–601.
- [3] R. H. Nochetto, A. Schmidt, C. Verdi, Adapting meshes and time-steps for phase change problems. (1997).
- [4] R. Rangarajan, A. J. Lew, Parameterization of planar curves immersed in triangulations with application to finite elements, *International Journal for Numerical Methods in Engineering* 88 (6) (2011) 556–585.
- [5] R. Rangarajan, A. J. Lew, Universal meshes: A method for triangulating planar curved domains immersed in nonconforming triangulations, *International Journal for Numerical Methods in Engineering* 98 (4) (2014) 236–264.
- [6] E. S. Gawlik, A. J. Lew, Unified analysis of finite element methods for problems with moving boundaries, (Preprint).
- [7] D. R. Lynch, Unified approach to simulation on deforming elements with application to phase change problems, *Journal of Computational Physics* 47 (1982) 387–411.
- [8] P. Lesaint, R. Touzani, Approximation of the heat equation in a variable domain with application to the Stefan problem, *SIAM Journal on Numerical Analysis* 26 (1989) 366–379.
- [9] C. W. Hirt, A. A. Amsden, J. L. Cook, An arbitrary Lagrangian-Eulerian computing method for all flow speeds, *Journal of Computational Physics* 14 (3) (1974) 227–253.
- [10] T. J. R. Hughes, W. K. Liu, T. K. Zimmermann, Lagrangian-Eulerian finite element formulation for incompressible viscous flows, *Computer methods in applied mechanics and engineering* 29 (3) (1981) 329–349.

- [11] J. Donea, S. Giuliani, J. P. Halleux, An arbitrary Lagrangian-Eulerian finite element method for transient dynamic fluid-structure interactions, *Computer Methods in Applied Mechanics and Engineering* 33 (1) (1982) 689–723.
- [12] M. Souli, A. Ouahsine, L. Lewin, ALE formulation for fluid-structure interaction problems, *Computer methods in applied mechanics and engineering* 190 (5) (2000) 659–675.
- [13] P. Geuzaine, C. Grandmont, C. Farhat, Design and analysis of ALE schemes with provable second-order time-accuracy for inviscid and viscous flow simulations, *Journal of Computational Physics* 191 (1) (2003) 206–227.
- [14] C. Farhat, P. Geuzaine, Design and analysis of robust ALE time-integrators for the solution of unsteady flow problems on moving grids, *Computer Methods in Applied Mechanics and Engineering* 193 (39) (2004) 4073–4095.
- [15] C. Farhat, K. G. van der Zee, P. Geuzaine, Provably second-order time-accurate loosely-coupled solution algorithms for transient nonlinear computational aeroelasticity, *Computer methods in applied mechanics and engineering* 195 (17) (2006) 1973–2001.
- [16] C. Farhat, A. Rallu, K. Wang, T. Belytschko, Robust and provably second-order explicit-explicit and implicit-explicit staggered time-integrators for highly non-linear compressible fluid-structure interaction problems, *International Journal for Numerical Methods in Engineering* 84 (1) (2010) 73–107.
- [17] N. Takashi, ALE finite element computations of fluid-structure interaction problems, *Computer methods in applied mechanics and engineering* 112 (1) (1994) 291–308.
- [18] W. K. Liu, H. Chang, C. J. Chen, T. Belytschko, Arbitrary Lagrangian-Eulerian Petrov-Galerkin finite elements for nonlinear continua, *Computer Methods in Applied Mechanics and Engineering* 68 (3) (1988) 259–310.
- [19] J. Wang, M. S. Gadala, Formulation and survey of ALE method in nonlinear solid mechanics, *Finite Elements in Analysis and Design* 24 (4) (1997) 253–269.
- [20] H. Askes, E. Kuhl, P. Steinmann, An ALE formulation based on spatial and material settings of continuum mechanics. Part 2: Classification and applications, *Computer Methods in Applied Mechanics and Engineering* 193 (39) (2004) 4223–4245.
- [21] A. R. Khoei, M. Anahid, K. Shahim, An extended arbitrary Lagrangian-Eulerian finite element method for large deformation of solid mechanics, *Finite Elements in Analysis and Design* 44 (6) (2008) 401–416.
- [22] J. M. Sullivan, D. R. Lynch, Finite element simulation of planar instabilities during solidification of an undercooled melt, *Journal of Computational Physics* 69 (1987) 81–111.
- [23] N. Zabaras, Y. Ruan, Moving and deforming finite-element simulation of two-dimensional Stefan problems, *Communications in Applied Numerical Methods* 6 (1990) 495–506.

- [24] M. R. Albert, K. O'Neill, Moving boundary-moving mesh analysis of phase change using finite elements with transfinite mappings, *International Journal for Numerical Methods in Engineering* 23 (1986) 591–607.
- [25] G. Beckett, J. A. Mackenzie, M. L. Robertson, A moving mesh finite element method for the solution of two-dimensional Stefan problems, *Journal of Computational Physics* 168 (2001) 500–518.
- [26] D. Boffi, L. Gastaldi, Stability and geometric conservation laws for ALE formulations, *Computer methods in applied mechanics and engineering* 193 (42) (2004) 4717–4739.
- [27] J. A. Mackenzie, W. R. Mekwi, An unconditionally stable second-order accurate ALE–FEM scheme for two-dimensional convection–diffusion problems, *IMA Journal of Numerical Analysis* 32 (3) (2012) 888–905.
- [28] L. Formaggia, F. Nobile, Stability analysis of second-order time accurate schemes for ALE–FEM, *Computer methods in applied mechanics and engineering* 193 (39) (2004) 4097–4116.
- [29] L. Formaggia, F. Nobile, A stability analysis for the arbitrary Lagrangian Eulerian formulation with finite elements, *East West Journal of Numerical Mathematics* 7 (1999) 105–132.
- [30] L. Gastaldi, A priori error estimates for the arbitrary Lagrangian Eulerian formulation with finite elements, *Journal of Numerical Mathematics* 9 (2) (2001) 123–156.
- [31] A. Bonito, I. Kyza, R. H. Nochetto, Time-discrete higher order ALE formulations: Stability, (Preprint).
- [32] A. Bonito, I. Kyza, R. H. Nochetto, Time-discrete higher order ALE formulations: A priori error analysis, (Preprint).
- [33] F. Aymone, J. Luis, Mesh motion techniques for the ale formulation in 3d large deformation problems, *International journal for numerical methods in engineering* 59 (14) (2004) 1879–1908.
- [34] P. Z. Bar-Yoseph, S. Mereu, S. Chippada, V. J. Kalro, Automatic monitoring of element shape quality in 2-D and 3-D computational mesh dynamics, *Computational Mechanics* 27 (5) (2001) 378–395.
- [35] A. Masud, Effects of mesh motion on the stability and convergence of ALE based formulations for moving boundary flows, *Computational Mechanics* 38 (4-5) (2006) 430–439.
- [36] J.-P. P. J. Donea, A. Huerta, A. Rodriguez-Ferran, *Encyclopedia of Computational Mechanics*, John Wiley and Sons, Ltd., New York, 2004, Ch. 14: Arbitrary Lagrangian-Eulerian Methods.
- [37] C. Farhat, C. Degand, B. Koobus, M. Lesoinne, Torsional springs for two-dimensional dynamic unstructured fluid meshes, *Computer methods in applied mechanics and engineering* 163 (1) (1998) 231–245.

- [38] A. A. Johnson, T. E. Tezduyar, Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces, *Computer methods in applied mechanics and engineering* 119 (1) (1994) 73–94.
- [39] B. T. Helenbrook, Mesh deformation using the biharmonic operator, *International journal for numerical methods in engineering* 56 (7) (2003) 1007–1021.
- [40] T. E. Tezduyar, S. Sathe, T. Cragin, B. Nanna, B. S. Conklin, J. Pausewang, M. Schwaab, Modelling of fluid–structure interactions with the space–time finite elements: Arterial fluid mechanics, *International Journal for Numerical Methods in Fluids* 54 (6-8) (2007) 901–922.
- [41] R. Bonnerot, P. Jamet, A second order finite element method for the one-dimensional Stefan problem, *International Journal for Numerical Methods in Engineering* 8 (1974) 811–820.
- [42] R. Bonnerot, P. Jamet, A third order accurate discontinuous finite element method for the one-dimensional Stefan problem, *Journal of Computational Physics* 32 (1979) 145–167.
- [43] P. Jamet, Galerkin-type approximations which are discontinuous in time for parabolic equations in a variable domain, *SIAM Journal on Numerical Analysis* 15 (1978) 912–928.
- [44] S. Rhebergen, B. Cockburn, Space-time hybridizable discontinuous galerkin method for the advection–diffusion equation on moving and deforming meshes, in: *The Courant–Friedrichs–Lewy (CFL) Condition*, Springer, 2013, pp. 45–63.
- [45] S. Rhebergen, B. Cockburn, A space–time hybridizable discontinuous galerkin method for incompressible flows on deforming domains, *Journal of Computational Physics* 231 (11) (2012) 4185–4204.
- [46] C. S. Peskin, The immersed boundary method, *Acta Numerica* 11 (0) (2002) 479–517.
- [47] R. J. Leveque, Z. Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, *SIAM Journal on Numerical Analysis* 31 (4) (1994) 1019–1044.
- [48] E. A. Fadlun, R. Verzicco, P. Orlandi, J. Mohd-Yusof, Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations, *Journal of Computational Physics* 161 (1) (2000) 35–60.
- [49] H. S. Udaykumar, R. Mittal, P. Rampunggoon, A. Khanna, A sharp interface cartesian grid method for simulating flows with complex moving boundaries, *Journal of Computational Physics* 174 (1) (2001) 345–380.
- [50] J. A. Sethian, *Level set methods and fast marching methods : evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, Cambridge University Press, Cambridge, 1999.

- [51] P. Zhao, J. C. Heinrich, Front-tracking finite element method for dendritic solidification, *Journal of Computational Physics* 173 (2001) 765–796.
- [52] S. Xu, Z. Wang, An immersed interface method for simulating the interaction of a fluid with moving boundaries, *Journal of Computational Physics* 216 (2006) 454–493.
- [53] N. Palle, J. Dantzig, An adaptive mesh refinement scheme for solidification problems, *Metallurgical and Materials Transactions* 27 (1996) 707–717.
- [54] R. H. Nochetto, M. Paolini, C. Verdi, An adaptive finite element method for two-phase Stefan problems in two space dimensions. Part I: Stability and error estimates, *Mathematics of Computation* 57 (1991) 73–108.
- [55] J. Baiges, R. Codina, The fixed-mesh ALE approach applied to solid mechanics and fluid–structure interaction problems, *International journal for numerical methods in engineering* 81 (12) (2010) 1529–1557.
- [56] J. Baiges, R. Codina, H. Coppola-Owen, The fixed-mesh ALE approach for the numerical simulation of floating solids, *International Journal for Numerical Methods in Fluids* 67 (8) (2011) 1004–1023.
- [57] G. M. Lieberman, *Second Order Parabolic Differential Equations*, World Scientific, Singapore, 1996.
- [58] A. Ern, J. L. Guermond, *Theory and Practice of Finite Elements*, Springer, New York, 2004.
- [59] K. Burrage, J. C. Butcher, F. H. Chipman, An implementation of singly-implicit Runge-Kutta methods, *BIT Numerical Mathematics* 20 (3) (1980) 326–340.
- [60] E. Hairer, G. Wanner, *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*, Springer, Berlin, 2002.
- [61] R. Rangarajan, A. J. Lew, Analysis of a method to parameterize planar curves immersed in triangulations, *SIAM Journal on Numerical Analysis* 51 (3) (2013) 1392–1420.
- [62] S. C. Brenner, L. R. Scott, *The Mathematical Theory of Finite Element Methods*, Springer, New York, 1994.
- [63] W. Ying, C. S. Henriquez, D. J. Rose, Composite backward differentiation formula: an extension of the TR-BDF2 scheme, Submitted to *Applied Numerical Mathematics*.

Appendix A. Singly Diagonally Implicit Runge Kutta time integrators.

Tables A.3–A.5 record the coefficients $\gamma > 0$ and $\beta_{ij} \in \mathbb{R}$, $i = 1, 2, \dots, s$, $j = 0, 1, \dots, i - 1$ for a collection of SDIRK methods (20) of orders 2 through 4.

Note that the structure of the Runge-Kutta stages in (20) differs from the structure that is most familiar to Runge-Kutta practitioners [60]. The former structure, which is algorithmically better-suited for problems with time-dependent mass matrices, is obtainable

Table A.3: Coefficients β_{ij} for a $s = 2$ -stage SDIRK scheme of order 2. ($\gamma = 1 - \sqrt{2}/2$)

$i \setminus j$	0	1
1	1	
2	$-\sqrt{2}$	$1 + \sqrt{2}$

Table A.4: Coefficients β_{ij} for a $s = 3$ -stage SDIRK scheme of order 3. ($\gamma = 0.43586652150845899942$)

$i \setminus j$	0	1	2
1	1.0000000000000000		
2	0.352859819860479140	0.647140180139520860	
3	-1.25097989505606042	3.72932966244456977	-1.47834976738850935

from any L-stable SDIRK scheme as follows. Let a_{ij} , b_j , and c_j , $i, j = 1, 2, \dots, s$, be the coefficients of an SDIRK scheme with Butcher tableaux

$$\begin{array}{c|cccc}
 c_1 & a_{11} & a_{12} & \cdots & a_{1s} \\
 c_2 & a_{21} & a_{22} & \cdots & a_{2s} \\
 \vdots & \vdots & \vdots & \ddots & \vdots \\
 c_s & a_{s1} & a_{s2} & \cdots & a_{ss} \\
 \hline
 & b_1 & b_2 & \cdots & b_s
 \end{array} \tag{A.1}$$

By definition, $a_{11} = a_{22} = \cdots = a_{ss}$ and $a_{ij} = 0$ for $j > i$. Assume that the scheme is L-stable, i.e. $b_j = a_{sj}$, $j = 1, 2, \dots, s$. Then the coefficients γ and β_{ij} in the formulation (20) are related to a_{ij} , b_j , and c_j via

$$\begin{aligned}
 \gamma &= a_{11} \\
 \beta_{ij} &= \begin{cases} \delta_{ij} - a_{ij}^* & \text{if } j > 0 \\ \sum_{k=1}^i a_{ik}^* & \text{if } j = 0. \end{cases}
 \end{aligned}$$

Here, δ_{ij} denotes the Kronecker delta and a_{ij}^* is the i, j entry of the matrix γA^{-1} , where $A = (a_{ij})$. The equivalence between (20) and the scheme defined by (A.1) is proven in [63].

Appendix B. The closest point projection onto a moving curve and its time derivative.

The following paragraphs derive explicit expressions for the time derivative of the closest point projection of a fixed point in space onto a moving curve. Such expressions are needed in numerical implementations for the evaluation of (29) when the boundary evolution operator $\gamma_h^{n,t}$ is given by (24).

Consider a moving curve $c^t \in \mathcal{C} := \{s \in C^2([0, 1], \mathbb{R}^2) \mid s'(\theta) \neq 0 \ \forall \theta \in [0, 1]\}$ whose velocity at any point $y = c^t(\theta) \in \text{image}(c^t)$ is given by $v^t(y) = \dot{c}^t(\theta)$. Let $\hat{n}^t(y)$, $\hat{t}^t(y)$, and $\kappa^t(y)$ denote the unit normal vector, unit tangent vector, and signed curvature at y ,

Table A.5: Coefficients β_{ij} for a $s = 5$ -stage SDIRK scheme of order 4. ($\gamma = 1/4$)

$i \setminus j$	0	1	2	3	4
1	1				
2	-1	2			
3	$-\frac{13}{25}$	$\frac{42}{25}$	$-\frac{4}{25}$		
4	$-\frac{4}{17}$	$\frac{89}{68}$	$-\frac{25}{136}$	$\frac{15}{136}$	
5	$\frac{7}{3}$	$-\frac{37}{12}$	$-\frac{103}{24}$	$\frac{275}{8}$	$-\frac{85}{3}$

respectively, and let π^t and ϕ^t denote the closest point projection onto $\text{image}(c^t)$ and the signed distance function on \mathbb{R}^2 , respectively, as in Section 5. Let τ denote the arclength parameter on $\text{image}(c^t)$. Henceforth, we employ the arclength parametrization and write $c^t(\tau)$ to denote the point on $\text{image}(c^t)$ with arclength parameter τ .

With respect to the arclength parametrization, the unit normal, unit tangent, and signed curvature satisfy the following relations at any point $y = c^t(\tau)$:

$$\hat{t}^t(y) = \frac{\partial c^t}{\partial \tau}(\tau), \quad \frac{\partial \hat{t}^t}{\partial \tau}(y) = \kappa^t(y) \hat{n}^t(y), \quad \frac{\partial \hat{n}^t}{\partial \tau}(y) = -\kappa^t(y) \hat{t}^t(y).$$

Here, for a given function $f^t : \text{image}(c^t) \rightarrow \mathbb{R}^k$, $k \in \{1, 2\}$, we are abusing notation by writing

$$\frac{\partial f^t}{\partial \tau}(y) := \left. \frac{\partial}{\partial \tau} \right|_t f^t(c^t(\tau))$$

for any $y = c^t(\tau) \in \text{image}(c^t)$. Likewise, we write

$$\frac{\partial g^t}{\partial t}(x) = \left. \frac{\partial}{\partial t} \right|_x g^t(x)$$

for a function $g^t : \mathbb{R}^2 \rightarrow \mathbb{R}^k$, $k \in \{1, 2\}$.

The closest point projection satisfies

$$x - \pi^t(x) = \phi^t(x) \hat{n}^t(\pi^t(x)) \tag{B.1}$$

for any $x \in \mathbb{R}^2$ for which $\pi^t(x)$ is uniquely defined. Another identity that will be of use momentarily concerns the normal velocity $v_n^t(y) := v^t(y) \cdot \hat{n}^t(y)$. Namely,

$$\frac{\partial v_n^t}{\partial \tau}(y) = \hat{n}^t(y) \cdot \frac{\partial v^t}{\partial \tau}(y) - \kappa^t(y) \hat{t}^t(y) \cdot v^t(y) \tag{B.2}$$

for any $y \in \text{image}(c^t)$ by the product rule.

Proposition. *Suppose $\{c^t\}_{t \in [0, T]} \subset \mathcal{C}$ is a family of curves such that the map*

$$c : \{(\tau, t) : 0 \leq \tau \leq \text{length}(\text{image}(c^t)), 0 \leq t \leq T\} \rightarrow \mathbb{R}^2 \\ (\tau, t) \mapsto c^t(\tau)$$

is of class C^2 . Let $x \in \mathbb{R}^2$ be a point for which $\pi^t(x)$ is uniquely defined and $\phi^t(x)\kappa^t(\pi^t(x)) < 1$ for every $0 \leq t \leq T$. Then

$$\frac{\partial \pi^t}{\partial t}(x) = v_n^t(\pi^t(x))\hat{n}^t(\pi^t(x)) + \sigma^t(x)\hat{t}^t(\pi^t(x)) \quad (\text{B.3})$$

for every $0 \leq t \leq T$, where

$$\sigma^t(x) = \frac{\phi^t(x)\frac{\partial v_n^t}{\partial \tau}(\pi^t(x))}{1 - \phi^t(x)\kappa^t(\pi^t(x))}. \quad (\text{B.4})$$

Proof. Let $\bar{\tau}^t(x)$ denote the arclength parameter along $\text{image}(c^t)$ assumed by $\pi^t(x)$; that is,

$$c^t(\bar{\tau}^t(x)) = \pi^t(x). \quad (\text{B.5})$$

Differentiating this relation with respect to time gives

$$v^t(\pi^t(x)) + \hat{t}^t(\pi^t(x))\frac{\partial \bar{\tau}^t}{\partial t}(x) = \frac{\partial \pi^t}{\partial t}(x). \quad (\text{B.6})$$

On the other hand, relation (B.1) implies that

$$(x - c^t(\bar{\tau}^t(x))) \cdot \hat{t}^t(c^t(\bar{\tau}^t(x))) = 0 \quad (\text{B.7})$$

for every t . Using the fact that $\hat{t}^t(c^t(\bar{\tau}^t(x))) = \frac{\partial c^t}{\partial \tau}(\bar{\tau}^t(x))$ has unit length, the time derivative of (B.7) reads

$$-v^t(\pi^t(x)) \cdot \hat{t}^t(\pi^t(x)) - \frac{\partial \bar{\tau}^t}{\partial t} + \phi^t(x)\hat{n}^t(\pi^t(x)) \cdot \left(\frac{\partial v^t}{\partial \tau}(\pi^t(x)) + \kappa^t(\pi^t(x))\hat{n}^t(\pi^t(x))\frac{\partial \bar{\tau}^t}{\partial t} \right) = 0. \quad (\text{B.8})$$

Together, relations (B.6) and (B.8) provide enough information to solve for the normal and tangential components of $\frac{\partial \pi^t}{\partial t}(x)$.

The normal component of $\frac{\partial \pi^t}{\partial t}(x)$ is obtained easily by dotting (B.6) with $\hat{n}^t(\pi^t(x))$, resulting in

$$\frac{\partial \pi^t}{\partial t}(x) \cdot \hat{n}^t(\pi^t(x)) = v_n^t(\pi^t(x)).$$

To compute the tangential component $\sigma^t(x) := \frac{\partial \pi^t}{\partial t} s(x) \cdot \hat{t}^t(\pi^t(x))$, take the dot product of (B.6) with $\hat{t}^t(\pi^t(x))$ and simplify (B.8) to obtain the following system of equations in two unknowns $\sigma^t(x)$ and $\frac{\partial \bar{\tau}^t}{\partial t}$:

$$\begin{aligned} v^t(\pi^t(x)) \cdot \hat{t}^t(\pi^t(x)) + \frac{\partial \bar{\tau}^t}{\partial t}(x) &= \sigma^t(x) \\ -v^t(\pi^t(x)) \cdot \hat{t}^t(\pi^t(x)) - \frac{\partial \bar{\tau}^t}{\partial t}(x) + \phi^t(x)\hat{n}^t(\pi^t(x)) \cdot \frac{\partial v^t}{\partial \tau}(\pi^t(x)) + \kappa^t(\pi^t(x))\phi^t(x)\frac{\partial \bar{\tau}^t}{\partial t}(x) &= 0. \end{aligned}$$

Solving this system and invoking (B.2) leads to (B.4). \square

Remark. The restriction $\phi^t(x)\kappa^t(\pi^t(x)) < 1$ in the preceding proposition is mild. In general, $\phi^t(x)\kappa^t(\pi^t(x)) \leq 1$ whenever $\pi^t(x)$ is uniquely defined. Indeed, since $|x - c^t(\tau)|^2$ is minimal

at $\tau = \bar{\tau}^t(x)$, it follows that

$$0 \leq \frac{\partial^2}{\partial \tau^2} \Big|_{\tau=\bar{\tau}^t(x)} |x - c^t(\tau)|^2 = 2(1 - \phi^t(x)\kappa^t(\pi^t(x))).$$

The assumption of strict inequality rules out degenerate cases in which $\frac{\partial^2}{\partial \tau^2} \Big|_{\tau=\bar{\tau}^t(x)} |x - c^t(\tau)|^2 = 0$.